Fabric Configuration User Manual

(2021.4 Version3.0)

Shenzhen Ziguang Tongchuang Electronics Co., Ltd.

Table of contents

List of figures

List of tables

1. Software Description

Fabric Configuration software is to configure the FPGA chip, and it is responsible for downloading the bit stream file generated by the user logic design to the FPGA chip.

Its main functions include:

- Supports the following FPGA chip types
  Titan series: PGT30G, PGT180H
  Logos series: PGL12G, PGL22G, PGL22GS, PGL25G, PGL50G, PGL50H, PGL100H
  Logos2 series: PG2L100H, PG2L25H
  Compact series: PGC1KG, PGC1KL, PGC2KG, PGC2KL, PGC4KD, PGC4KL, PGC4KLS, PGC7KD, PGC10KD
  Titan2 series: PG2T390H
- Support the following SPI serial Flash chip types

  Micron : M25P128, M25P64, M25P32, M25P16, N25Q32, N25Q64, N25Q128, N25Q256, N25Q512

  GigaDevice: GD25Q80C, GD25Q32C, GD25Q64C, GD25Q128C, GD25Q256D, GD25Q512C, MD25Q80C

  WinBond: W25Q40CL, W25Q80, W25Q16, W25Q32, W25Q64Q, W25Q128, W25Q256, W25Q512

  XMC : XM25QH16B, XM25QH32B, XM25QH64A, XM25QH128A, XM25QH256B

  SSMEC : SM25QH256M

  CYPRESS/Spansion: S25FL64, S25FL128, S25FL256L, S25FL256S, S25FL512

  ZBIT: ZB25VQ16, ZB25VQ32

  MXIC: MX25L32, MX25L64, MX25L128, MX25L256, MX25L512

  ISSI: IS25LP16D, IS25LP32D, IS25LP64D, IS25LP128D, IS25LP256D, IS25LP512D, IS25WP16D, IS25WP32D, IS25WP64D, IS25WP128D, IS25WP256D, IS25WP512D

  ESMT: EN25Q80C

  ADESTO: AT45DB081E

  FENTECH: FH25VQ80D
- Support Pango USB Cable I, Pango USB Cable II, Pango Parallel Cable I download cables
- Support FPGA ID, user code and status register read function, logic bit stream download, read back and check
- Support SPI Flash download, delete, read back and verify functions

2. Download Cable Introduction

2.1 USB download cable

To use the USB download cable, you need to install the corresponding driver. For the installation steps, refer to the introduction of the software installation section in the document Pango Design Suite Windows Install Guide. After the driver is successfully installed, insert the USB download cable. If there is a new device " Programming cable s " in the device manager, it means that the download cable can be used normally.

Pango USB Cable I and Pango USB Cable II are supported. The USB download cable is responsible for converting the USB signal of the PC into the JTAG signal or SPI signal required by the device. The physical connection of the download cable is subject to the instruction document provided in the download cable box, and its connection method is shown in Figure 2 -2 :



Figure 2 -1 USB Cable I download cable connection diagram

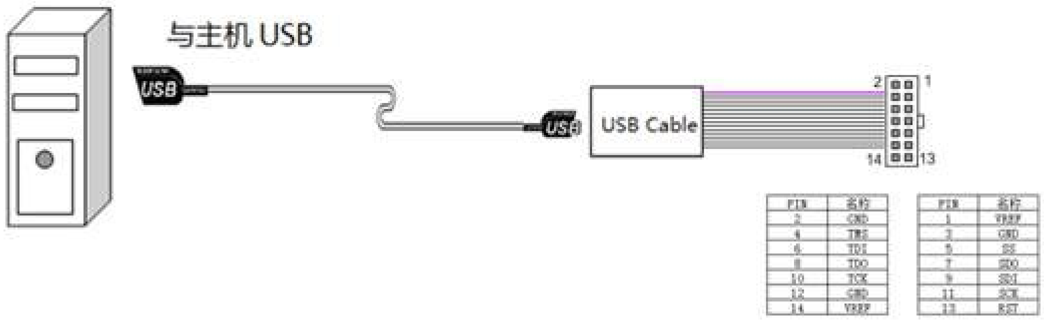


Figure 2 -2 USB Cable II download cable connection diagram

Note: In the 14PIN connector in the figure, the VREF of PIN1 supplies power to the JTAG interface, and the VREF of PIN14 supplies power to the SPI interface, which can be different.

2.2 Parallel download cable

To use the parallel port download cable, first make sure that there is an integrated parallel port card on the motherboard of the PC, and enable it in the BIOS. If you want to use the ECP mode, you also need to enable the parallel port ECP mode in the BIOS. Next, install the parallel port driver. For the installation steps, refer to the introduction of the software installation section in the document Pango Design Suite Windows Install Guide.

Currently it supports Pango Parallel Cable I. The parallel port download cable is responsible for converting the parallel signal of the PC into the JTAG signal or SPI signal required by the device. The physical connection of the download cable is subject to the instruction document provided in the download cable box, and the connection method is shown in the figure below:

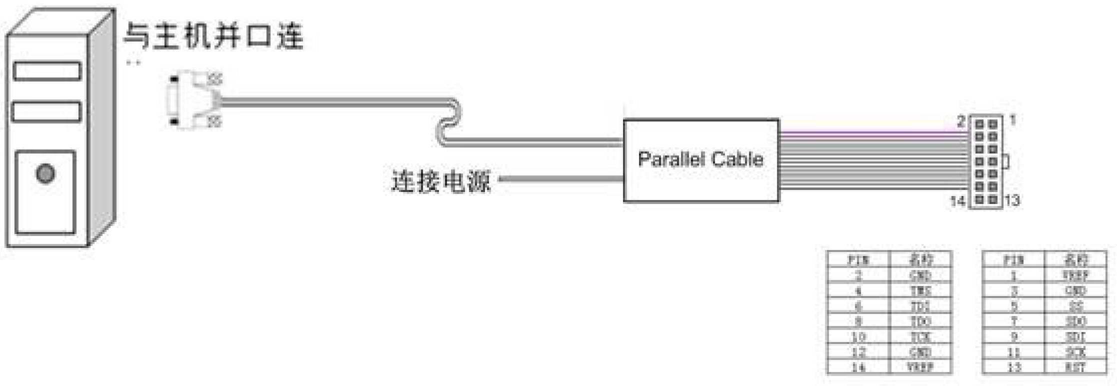


Figure 2.3 Parallel port download cable connection diagram

Note: In the 14PIN connector in the figure, the VREF of PIN1 supplies power to the JTAG interface, and the VREF of PIN14 supplies power to the SPI interface, which can be different.

2.3 Light status of the download cable

Both Pango USB Cable II and Pango Parallel Cable I have an indicator light that can display red and green. Pango USB Cable I has no indicator light function. It is defined that each read and write to FPGA and Flash is an operation. The display rules of the indicator lights are as follows :

a) In the initial state, the red light is always on, and the green light is off

b) The green light flashes while the operation is in progress, and the red light goes out

c) When the operation result is wrong, the red light is always on and the green light is off

d) The operation result is normal, the green light is always on, and the red light is off

3. Getting started with the software

First, a simple example is used to demonstrate how the Fabric Configuration software downloads the logical bit stream. The FPGA chip used in this example is PGL50H of Pango Logos series, and the logical bit stream file to be configured is named pgl50h.sbit.

3.1 Connect to Jtag Server

The Fabric Configuration software is paired with the corresponding Jtag Server software, and the versions between the two must be consistent, otherwise it will not work normally. After starting the Fabric Configuration software, click the button on the toolbar to pop up a wizard for connecting to the Jtag Server Cable, and follow the instructions of the wizard to connect to the server. This software uses TCP/IP protocol to communicate with the server, so it is necessary to specify the IP address and port. The IP address can be specified as the local IP address or the remote IP address.

When specifying as the local IP address, since the default address is the address of the local machine, just click to connect directly. If it is specified as a remote IP address, you need to start Jtag Server remotely first, find the software named cdt_js in the bin directory of the installation directory, and double-click to start it; users can also use the command line to start it. The specific startup method is: switch to The directory where the executable program is located, and then start the command line terminal of the operating system, and enter the command cdt_js –port 65420. For more parameters, you can enter the command cdt_js –help to view.

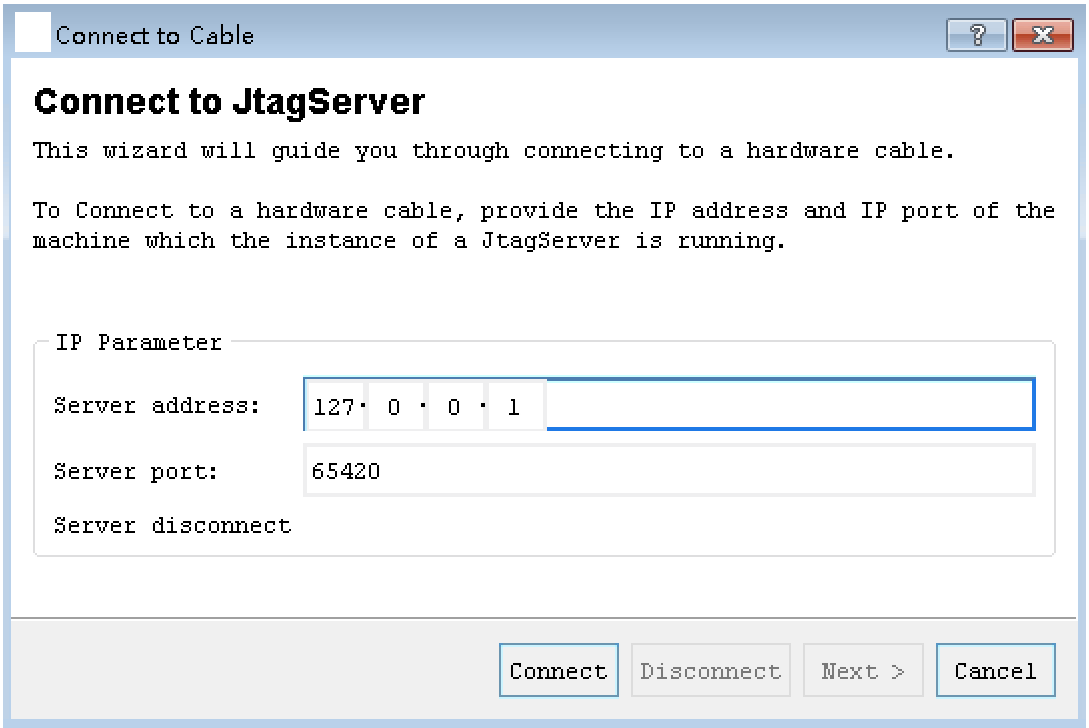


Figure 3 -1 Connect to Jtag Server

3.2 Set Cable parameters

After successfully connecting to the Jtag server, the connection interface will list all currently available Cables. You can click to select a Cable and set the TCK frequency. The first Cable is selected by default.

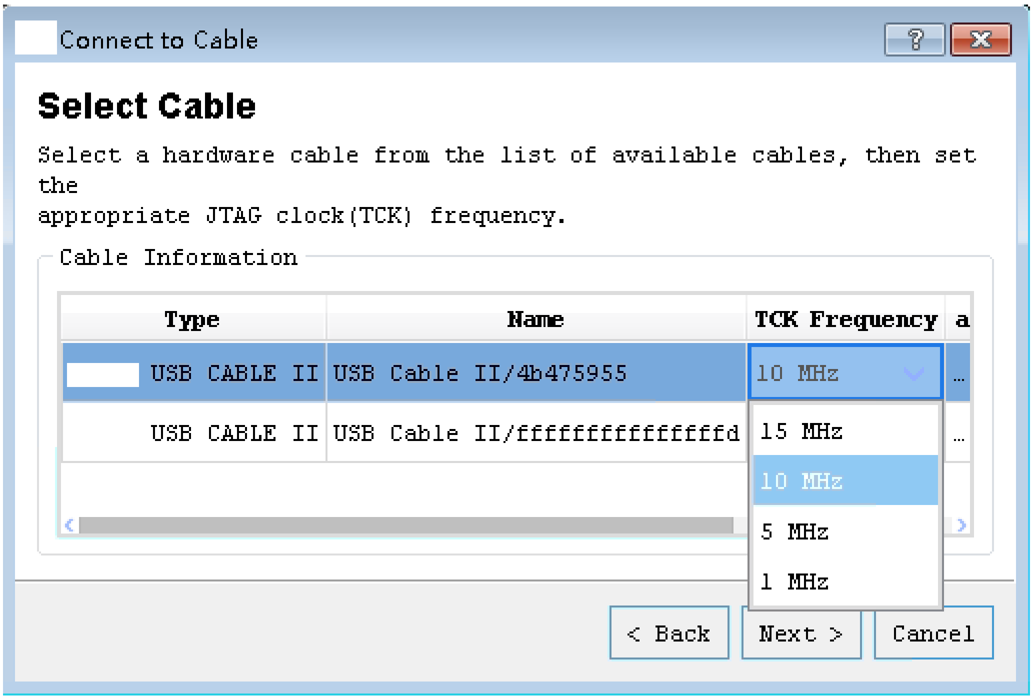


Figure 3 -2 Select Cable parameters

After selecting the required Cable, click the Next button to jump to the summary page, which will list the server connection information and Cable parameters. At this time, the cable parameter settings have not yet taken effect. Click Back to go back to the previous step and re-select. Click Cancel You can cancel this setting, and click Next to send the configuration information to the Jtag Server and set it up.

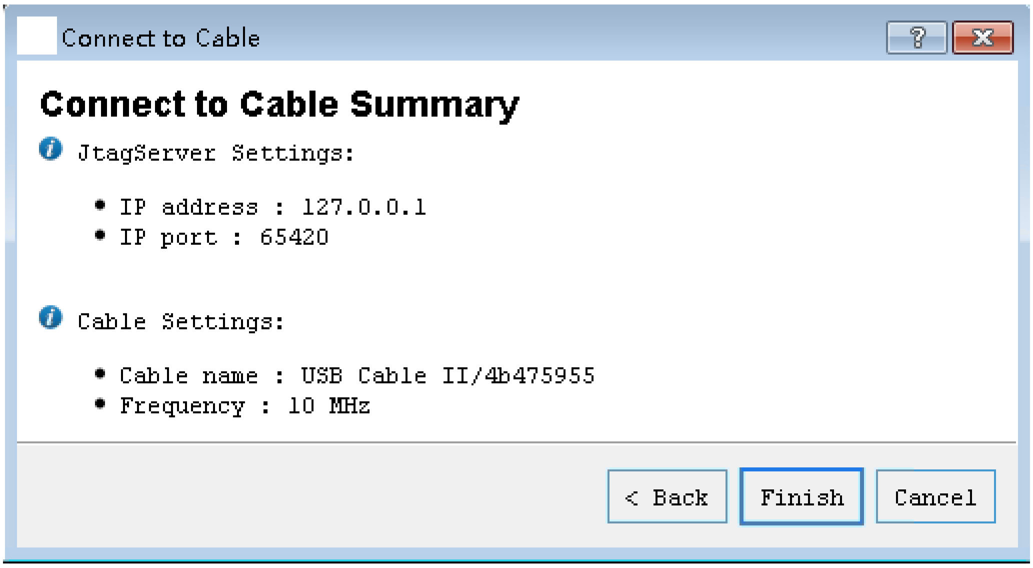


Figure 3 -3 Set Cable parameters

3.3 Use multiple cables at the same time

To use multiple cables on one machine at the same time, you can do it through the following steps (here we use two Cables at the same time as an example):

o Connect the two cables to the USB port, connect the corresponding circuit board, and turn on the power

o Switch to the bin folder of the PDS software installation directory (subsequent operations are performed in this directory)

o Start the command line terminal and enter the command cdt_js –port 65420

o Start the command line terminal and enter the command cdt_js –port 65421

o Start the Fabric Configuration software, click the button, and set the Server Port to 65420 in the wizard connecting to Jtag Server, select the first Cable, and leave the rest of the parameters default, and complete the setting of the first Cable

o Start the Fabric Configuration software, click the button, set the Server Port to 65421 in the wizard connected to Jtag Server, select the second Cable, and leave the rest of the parameters default, and complete the setting of the second Cable

o Test the sweep chain of two Cables separately, and configure the function

3.4 Scan the JTAG chain

Configuration Mode window on the left side of the main interface, click "Boundary Scan", and then right-click in the work area, a menu as shown in Figure 3 -and 4 will pop up, left-click "Scan Device" to execute the initialization chain operate.

Note : After opening the Fabric Configuration software, you can directly scan the Jtag chain. The software itself will automatically trigger the operation of connecting to the Jtag Server. The default IP address of the Jtag Server is 127.0.0.1, and the port number is 65420. After the connection is successful, the version will be checked, after the version check is successful, the Jtag chain operation will be scanned.

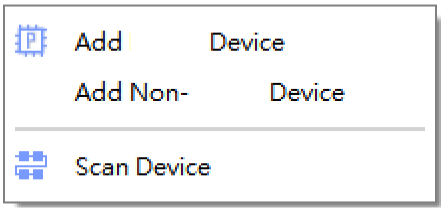


Figure 3 -4 workspace right mouse click menu

If the initialization chain is successful, all the devices scanned on the chain will be displayed in the work area, and the device information of the current device will be displayed in the device property window, and a dialog box will pop up to enable the user to add configuration files for the device, as shown in -Figure 3.5 Show:

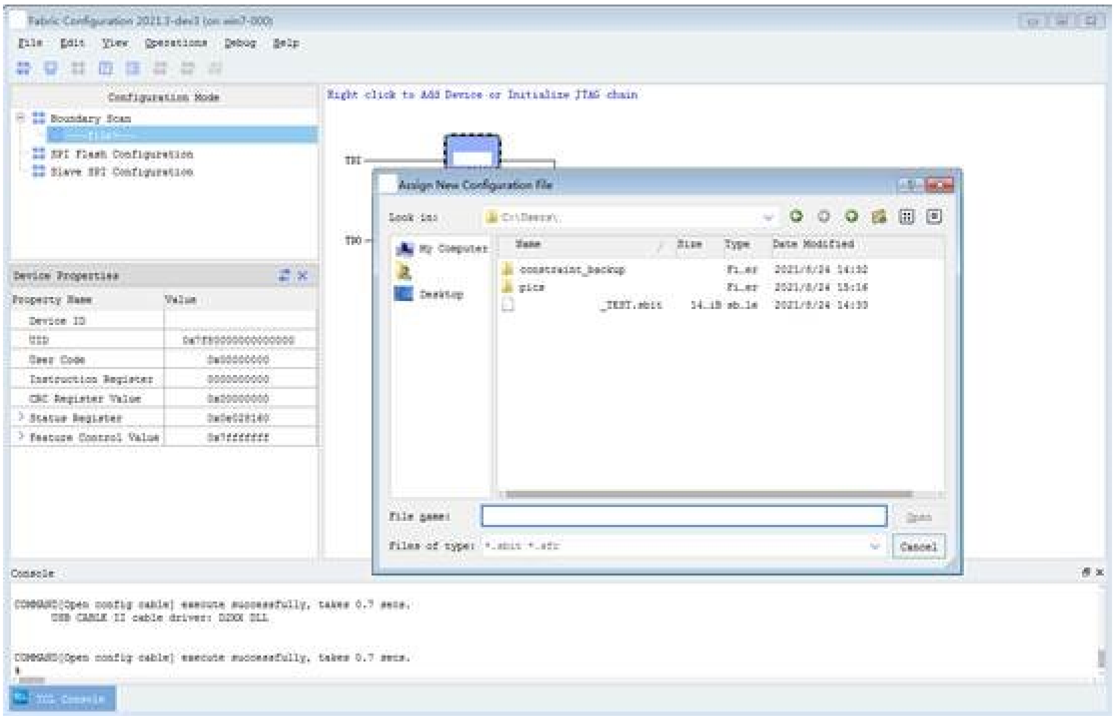


Figure 3-5 initialization chain successful

3.5 Configure Logical Bitstream File

Select the target bit stream file in the file configuration dialog box, add the configuration file, prompt the absolute path of the loaded file and display it in the information column, as shown in Figure 3 -6 :

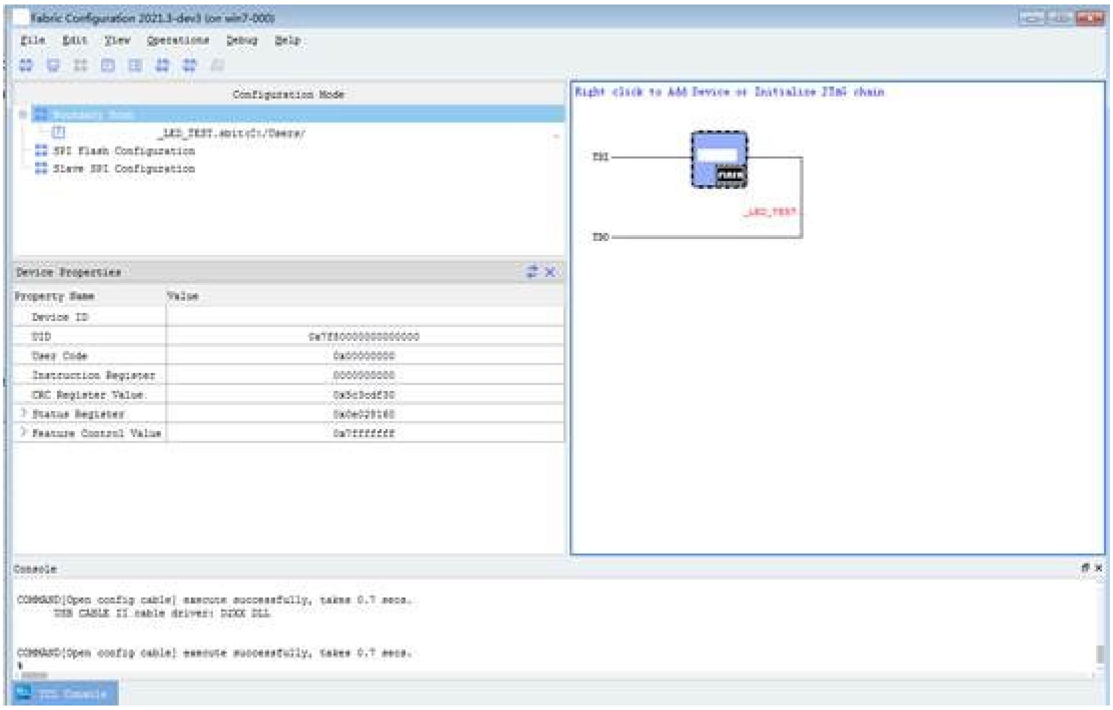


Figure 3 -6 Add configuration files for devices on the JTAG chain

3.6 Programming logic bitstream file

After the logic bitstream file is configured, it can be downloaded to the device. Before performing the download operation, make sure that you have connected to the Jtag Server and configured the bitstream file. After satisfying the downloading conditions, first select the FPGA device detected by the initialization chain, and then click the right mouse button on the FPGA, a menu as shown in Figure 3 -7 will pop up, click the "Program" operation with the left mouse button, The software starts to load the configured bit stream file pgl50h.sbit into the PGL50H device.

After the download progress bar is displayed, you can observe whether the DONE light on the board is on. If it is on, it means that the bit stream has been successfully downloaded and run; or check the done bit in the status register value through the Status Register column of the Device Properties window. If done A bit of 1 indicates that the bitstream was successfully downloaded and run. After the download is successful, the information in the device properties window is refreshed, as shown in Figure -38.

Note: For the detailed description of the status register, please refer to Chapter 5 Appendix -> Section 2 The functions indicated by each bit of the status register

Note: For some devices, the done pin may be multiplexed into other functions, and the status register can only be used to judge whether the download is successful.

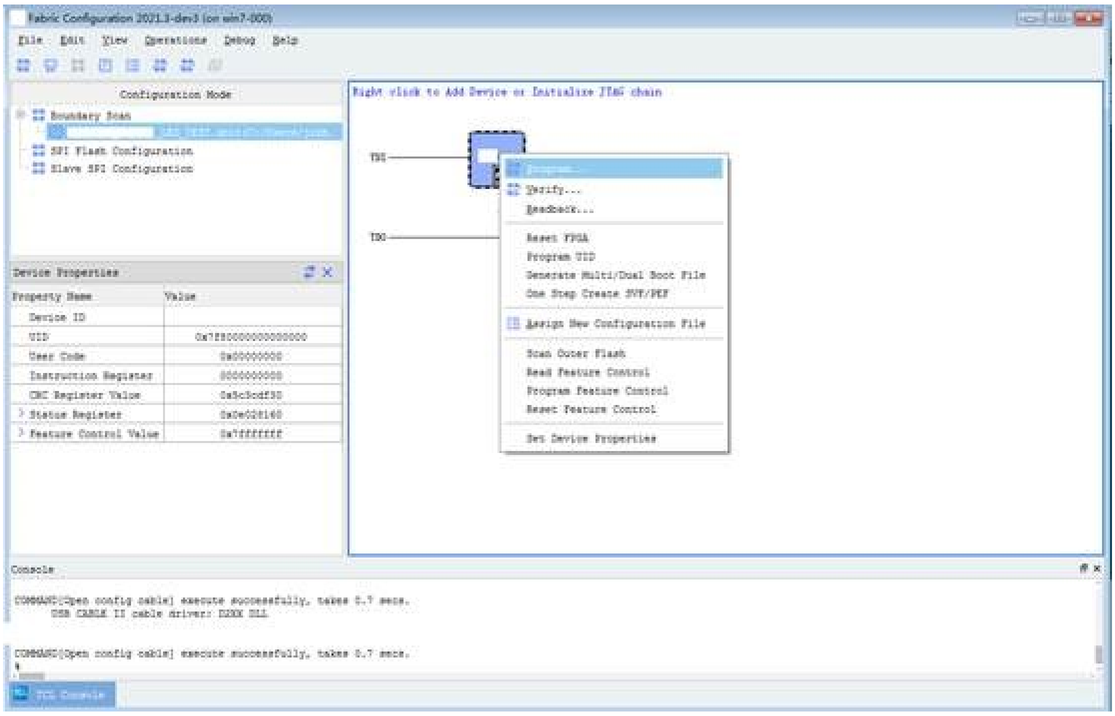


Figure 3-7 Right-click menu on FPGA

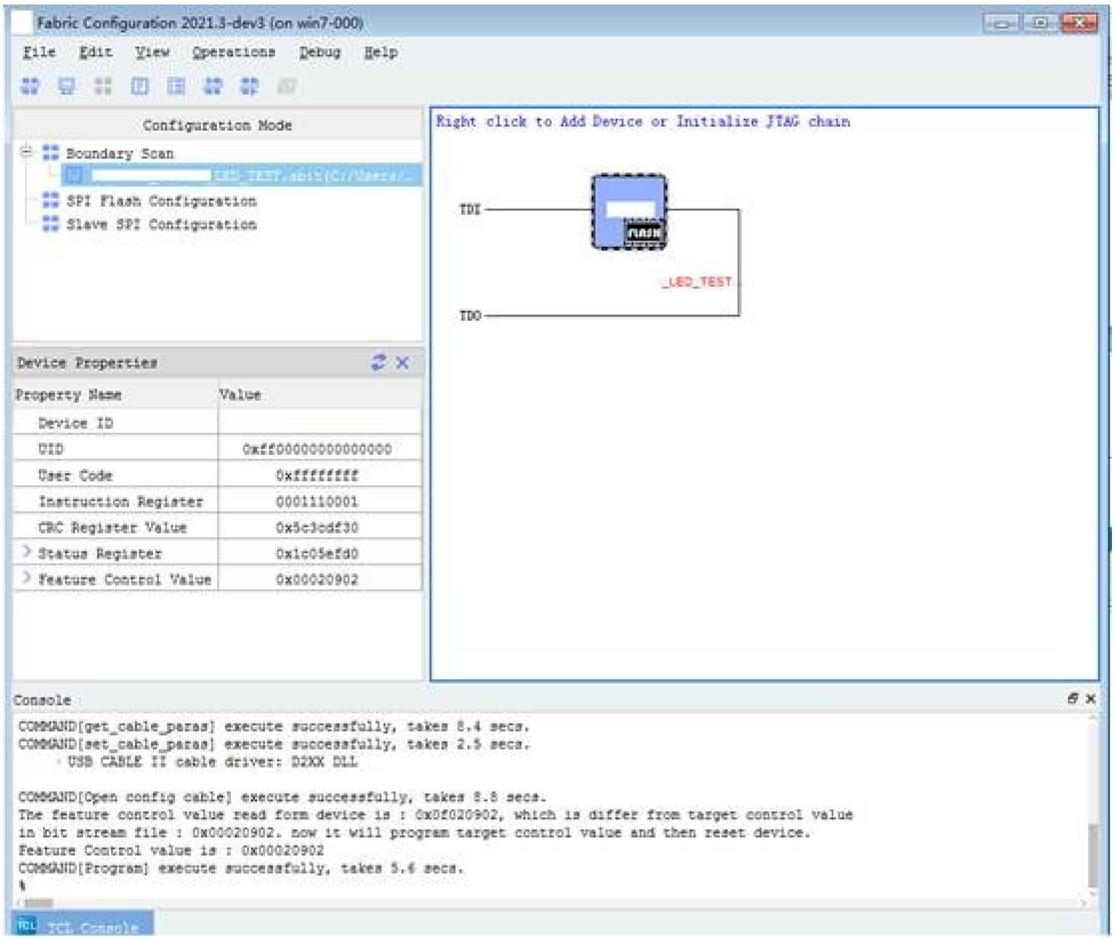


Figure 3-8 Download configuration file successfully

3.7 Verify logical bitstream file

After the operation of downloading the bit stream file is completed, in order to ensure that there is no error in the downloaded bit stream file during the download process, it can be verified by a check operation. This operation mainly compares the information in the configured bit stream file with the logic information downloaded to the FPGA device. If the two are consistent, it means that the verification is successful, otherwise the verification fails. First select the FPGA device that needs to be verified, then click the right mouse button on the FPGA, a menu as shown in Figure 3 7 will pop up -, and then click "Verify" with the left mouse button, the software will start to verify, verify After success, as shown in Figure 3 -9 :

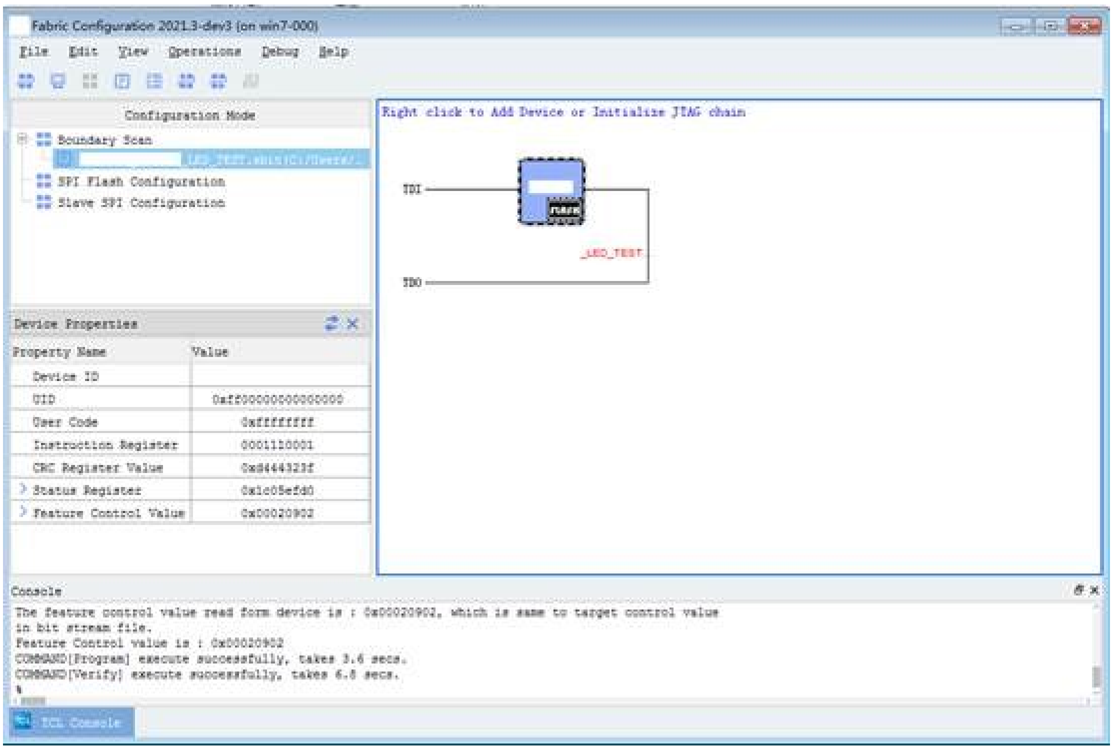


Figure 3 -9 Verify the downloaded configuration file

4. Fabric Configuration software description

4.1 User Interface Introduction

4.1.1 main user interface

on the toolbar of the PDS interface to start the Fabric Configuration software, or double-click "cdt_cfg.exe" in the bin directory of the PDS software installation directory to start the software, and the user can also start it on the shell terminal of the operating system and switch to the executable program For the directory where it is located, just enter the command cdt_cfg.exe. For more parameters, you can enter the command cdt_cfg.exe –help to view. After the software is started, it is shown in Figure 4 -1 :

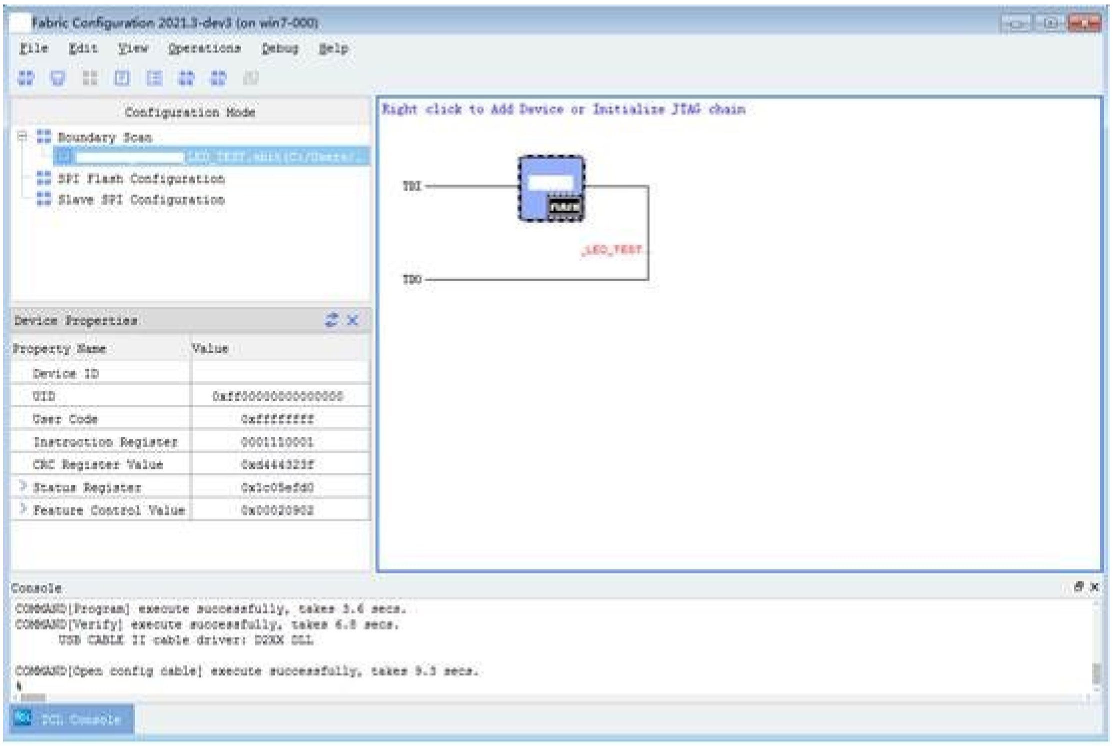


Figure 4 -1 Fabric Configuration main interface

- Menu bar: mainly includes 6 drop - down menus: File, Edit, View, Operations, Debug and Help. Its use method is similar to common Windows software.
- Toolbar: mainly includes shortcut buttons for common functions, which is convenient for users to operate.
- Work area: display devices scanned on the JTAG chain, or devices manually added by the user.
- Information display area: Display processing information in Fabric Configuration, such as operation step information, warning information and error information, etc.
- Device property window: display the basic property information of the device, such as ID, instruction register value, status register value, etc.

4.2 Menu Bar and Toolbar

4.2.1 File menu

As shown in Figure 4 -2, this menu mainly involves communication with Jtag Server and scanning devices.

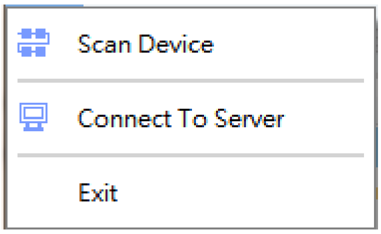


Figure 4 -2 File menu

【Scan Device 】: used to detect and initialize the device, if the Boundary Scan mode is selected in the Configuration Mode window, the scan All devices on the JTAG chain If the JTAG detection of the configuration circuit of the device is correct, all devices scanned on the chain will be listed in the work area; if the SPI Flash Configuration mode is selected in the Configuration Mode window, the connected SPI devices will be scanned and the The device is listed in the workspace.

【 Connect To Server 】: Perform JTAG Server connection operation.

【 Exit 】: Used to exit the Fabric Configuration software.

4.2.2 Edit menu

in Figure 4 -3 :

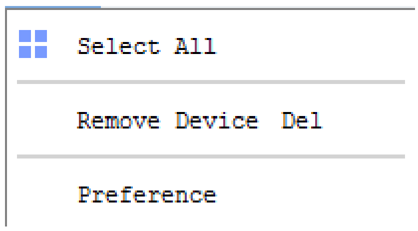


Figure 4 -3 Edit menu

【 Select All 】: Select all devices in the work area.

【 Remove Device 】: It is used to delete the selected device from the JTAG chain displayed on the interface, and the shortcut key is "Delete".

【 Preference 】: Set user-defined information and display it in the title bar to identify the characteristics of the current software; control whether to display the CRC value and Efuse value in the device property window, whether to modify the idcode (pgc) by default and whether to generate pef by default document.

4.2.3 View menu

as shown in the Figure 4 -4 shows:

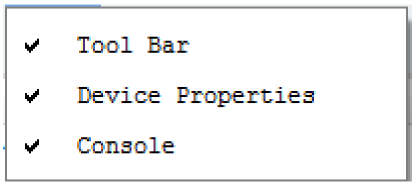


Figure 4 -4 ViewMenu

【 Tool Bar 】: The user can set the display and hide of the toolbar.

【 Device Properties 】: The user can set the display and hide of the device properties window.

【 Console 】: The user sets the display and hiding of the Console window.

4.2.4 Operations menu

in Figure 4 -5 :

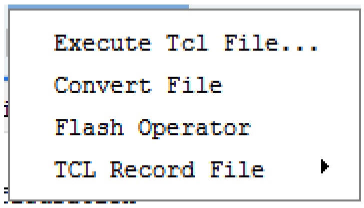


Figure 4 -5 OperationsMenu

【 Execute Tcl File 】: Used to execute Tcl files.

【Convert File 】: used for file conversion, mainly includes: ① convert FPGA bit stream file into Flash download file sfc format; ② use for synthesizing cascade data stream; Multi-functional combined data flow, main and self-loading dual boot data flow; ④ used to generate cascaded svf/pef files.

【 Flash Operator 】: Used to add user FLASH.

【 TCL Record File 】: It is used to record Tcl commands and generate user-specified Tcl scripts (including two ways of creating and attaching TCL files).

4.2.5 Debug menu

As shown in Figure 4 -6

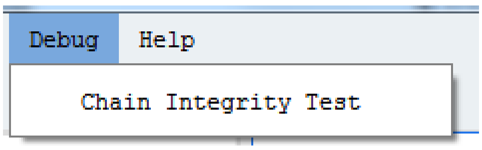


Figure 4 -6 Debug menu

【 Chain Integrity Test 】: JTAG chain integrity test, to test whether the current JTAG chain is a path.

4.2.6 Help menu

As shown in Figure 4 -7



Figure 4 -7 Help menu

【 About 】: Used to display Fabric Configuration version information.

【 Show User Help 】: It is used to open the Configuration user help document.

【 Help Topics 】: It is used to open the help information of Pango Assistance.

4.2.7 Basic operation of the toolbar

As shown in Figure 4 -8



Figure 4 -8 toolbar

【 Scan Device]: Corresponds to File-> Scan Device in the menu bar.

【 Connect To Server 】: Corresponding menu bar File->Connect To Server.

【 Select All 】: Corresponds to Edit->Select All in the menu bar.

【 Assign New Configuration File 】: Configure the logic bit stream file for FPGA device or SPI device.

【 Add Pango Device 】: Add FPGA device by selecting the logic bit stream file.

【 Program 】: Program and download the selected FPGA device or SPI Flash device.

【 Verify 】: Verify the selected FPGA device or SPI Flash device.

【 Erase 】: Erase the selected Flash device.

4.2.8 Device Properties window

As shown in Figure 49 -, the device property window is used to display the basic property information of the device, such as ID, status register value and so on. The user will update these attribute information in real time after each operation of the software.

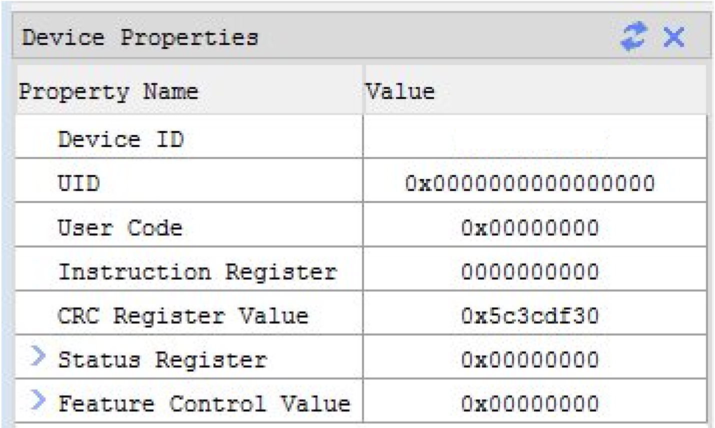


Figure 4-9 Device Properties window

【 Device ID 】: Display the unique identification code of the FPGA device.

【 UID 】: Display the serial code of the FPGA device.

【 User Code 】: Display the logical version or function information of the user's current bit stream.

【 Instruction Register 】: Display the instruction register value of the current state of the FPGA.

【 CRC Register Value 】: Display the CRC value of the current FPGA configuration stream.

【 Status Register 】: Display the value of FPGA current status register.

Note: For the detailed description of the instruction register, please refer to Chapter 5 Appendix -> Section 1 The functions indicated by each bit of the instruction register

Note: For the detailed description of the status register, please refer to Chapter 5 Appendix -> Section 2 The functions indicated by each bit of the status register

4.3 Function Description

4.3.1 Jtag boundary scan add device

Configuration Mode window on the left side of the main interface, select Boundary Scan mode, and then click the right mouse button in the work area, a menu as shown in Figure 4 -10 will pop up, left-click Scan Device, or left-click on the File menu or toolbar Clicking Scan Device will execute the initialization chain operation.

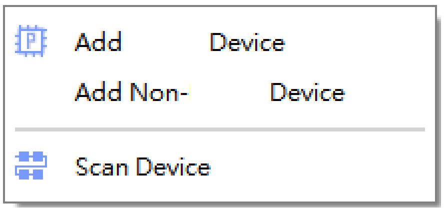


Figure 4-10 workspace right mouse click menu

If the initialization chain is successful, all devices scanned on the chain will be listed in the work area, and the corresponding device model will be displayed under the device in the work area, and a dialog box will pop up to enable the user to add configuration files for the device, as shown in Figure 4 -11 Show:

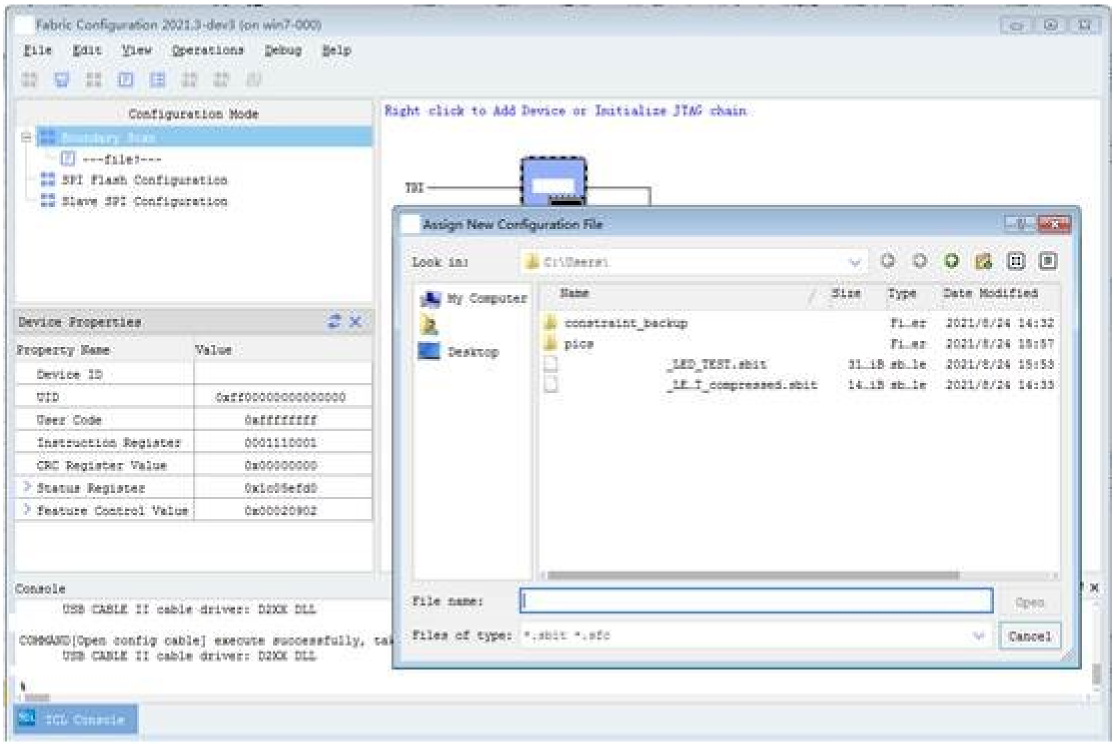


Figure 4 -11 Initialize the chain and add the.sbit file

If the selected configuration file matches the device detected by the initialization chain, the configuration file will be added, and the absolute path of the loaded file will be prompted in the information column, as shown in Figure 4 12 (PGC devices -can add sfc files by scanning the chain, if the sfc file is configured, only the embedded Flash can be operated, and the cram cannot be operated). If the selected configuration file does not match the device detected by the initialization chain (the sbit file does not match the actual device), Fabric Configuration will prevent the file from being added, and a warning will pop up to prompt the user, as shown in Figure 4 -13. If the initialization chain is unsuccessful, the pop-up dialog box is shown in Figure 414 -. There are many reasons for the failure of the initialization chain, and the solutions are generally: first, check whether the USB download cable is correctly connected to the host; second, check whether the USB driver of the download cable is installed correctly; third, check whether the board where the FPGA chip is located is working normally, and check whether the JTAG chain of the configuration circuit is complete; finally, try to restart the Fabric JtagServer software.

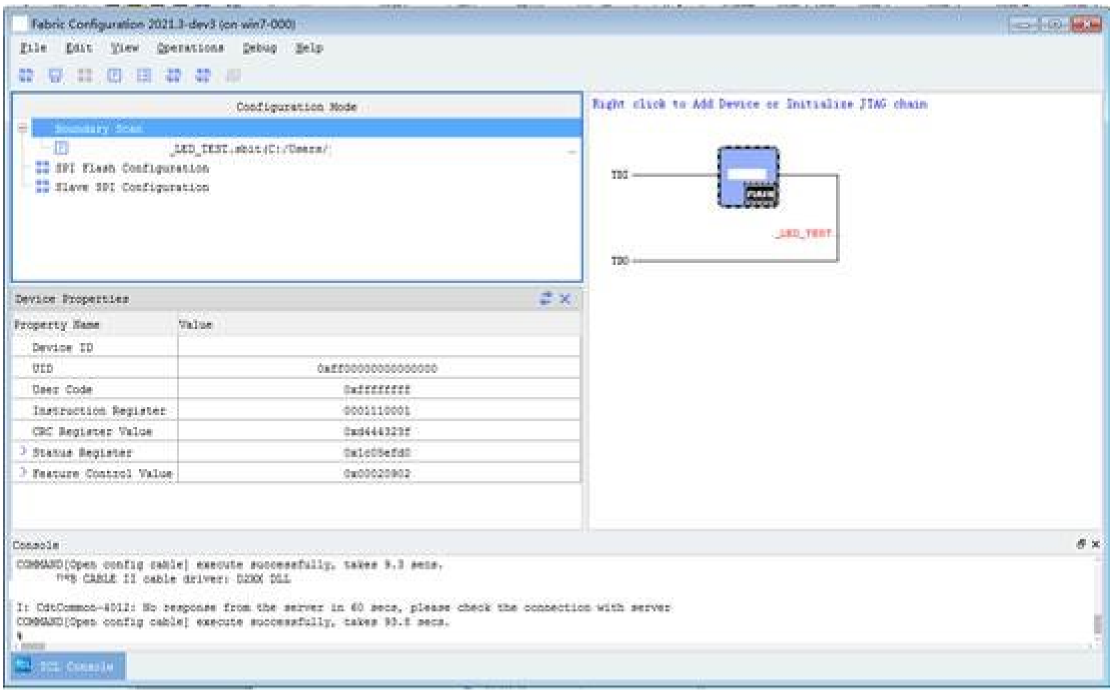


Figure 4-12 Configure the correct bitstream file

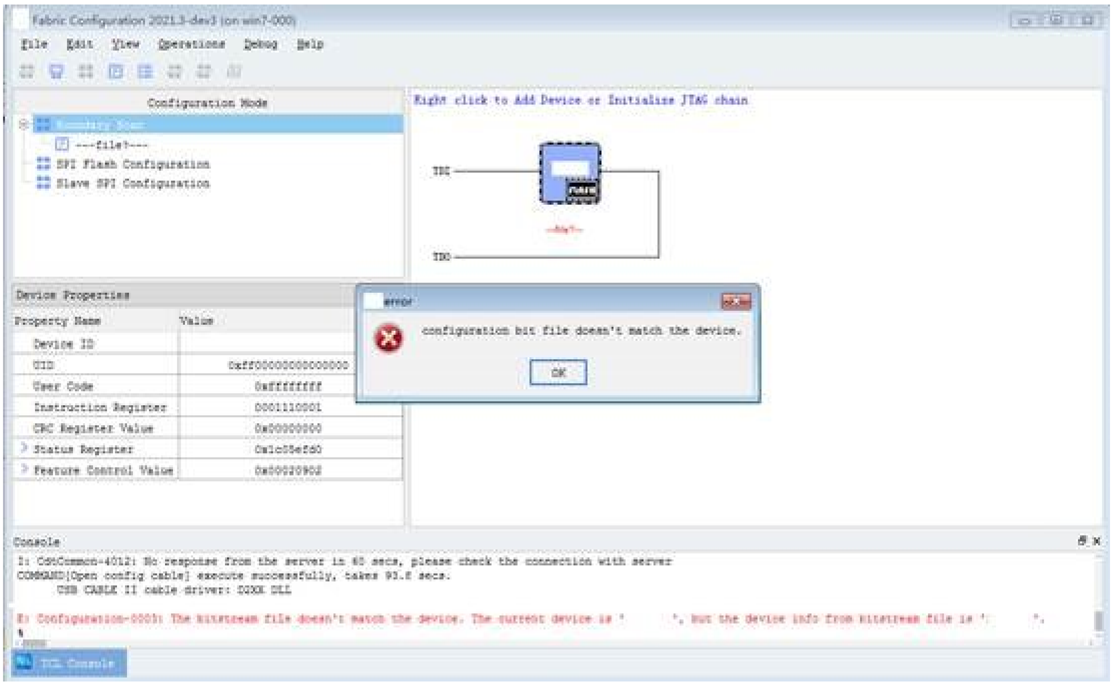


Figure 4-13 Misconfigured bitstream file

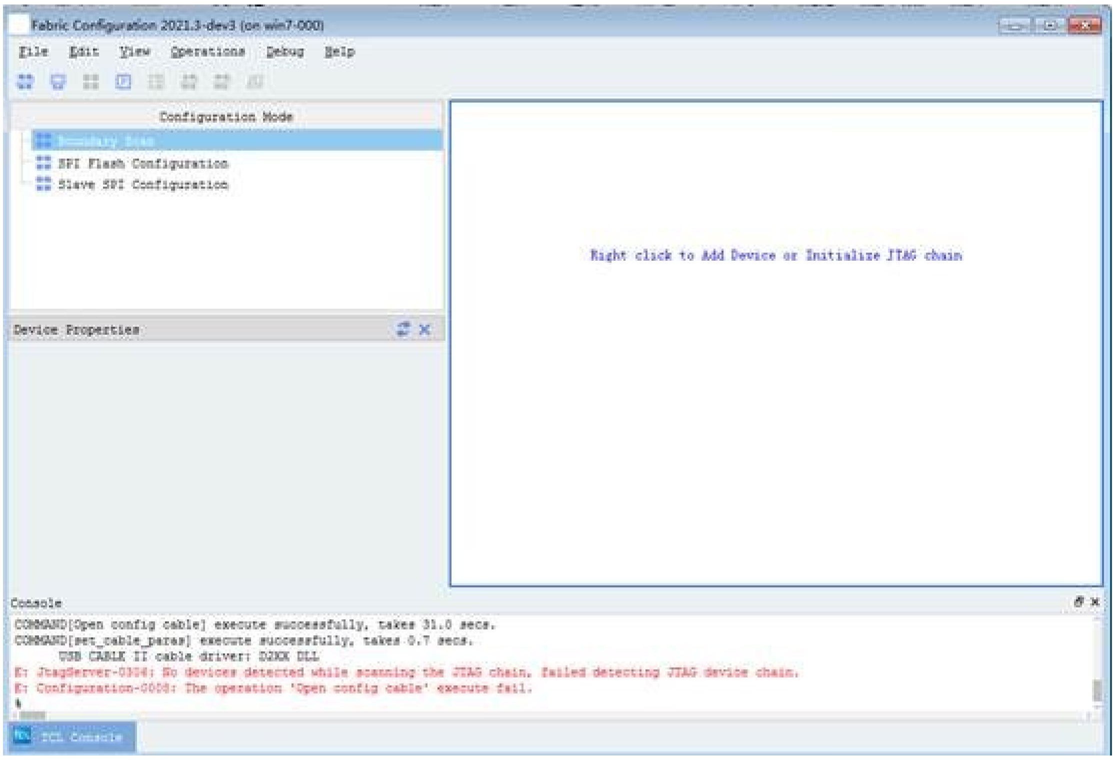


Figure 4 -14 Failed to initialize the chain

4.3.2 Parse target bitstream to add device

In addition to scanning the Jtag chain, users can also manually add FPGA devices by selecting a logic bitstream file. The specific operation is: right-click the mouse in the work area, and the menu shown in Figure 4 -10 will pop up, left-click "Add Pango Device", or left-click "Add Pango Device" in the toolbar to execute adding devices operation. If there is already a device in the chain, you need to click the left button where you want to add the device when adding it, and then right click to select Add Device, which will add the device to the location specified by JTAG.

If the device corresponding to the added bit stream file is a device supported by Fabric Configuration, add it to the workspace, which is the same as Figure 4 -12 where the boundary scan chain is successfully initialized ; if the configuration file is not recognized, it will prompt as shown in Figure 4 -15 shows:

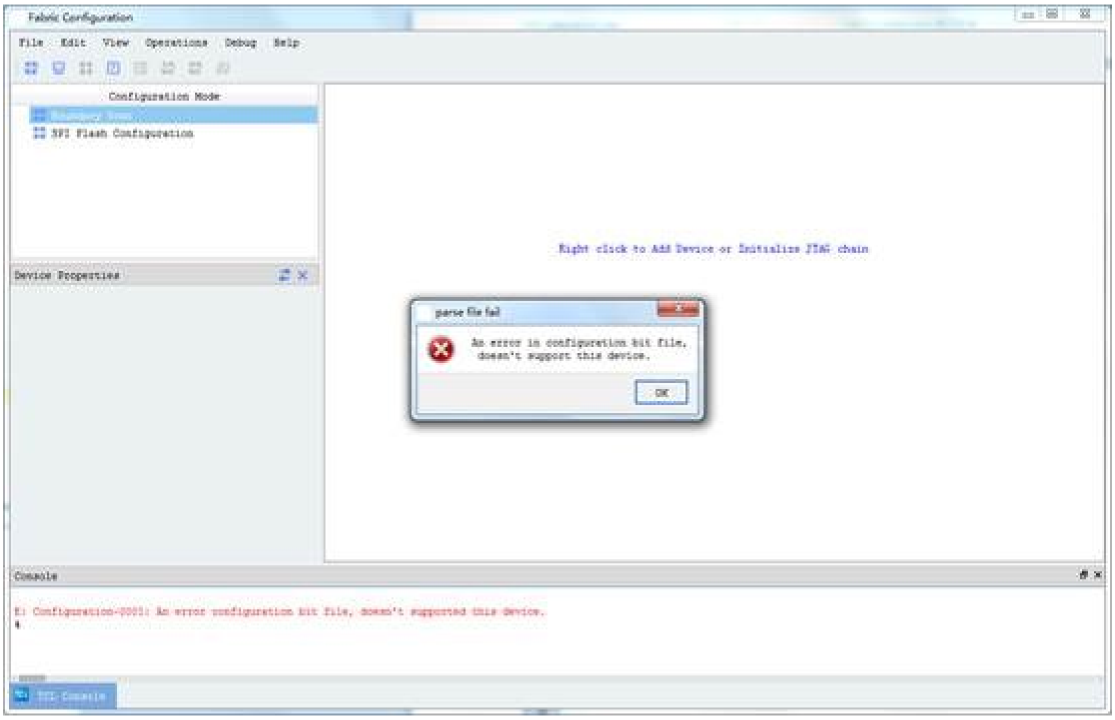


Figure 4 -15 Add bitstream files for unsupported devices

4.3.3 Configure a new bitstream file

if the user wants to re-add a configuration file for the device on the JTAG chain. The user first selects the device that needs to re-add the configuration file, right-clicks on the device, and the menu shown in Figure 4 -16 will pop up, and then left-clicks "Assign New Configuration File", or left-clicks " "Assign New Configuration File" can perform the operation of re-adding the configuration file.

If the execution is successful, the original configuration file will be replaced; if it fails, it means that the configuration file does not match the device detected on the JTAG chain at present, and a dialog box as shown in Figure 4 -17 will pop up.

Note: The Configuration software has the following restrictions on the names and file paths of all files:

1. Folder name: only letters and numbers are allowed Underscore (_) dash (-) dot (.) @ and space ( ), but spaces cannot appear at the beginning or end of the pathname.

2. File name: only letters and numbers are allowed Underscore (_) dash (-) dot (.).

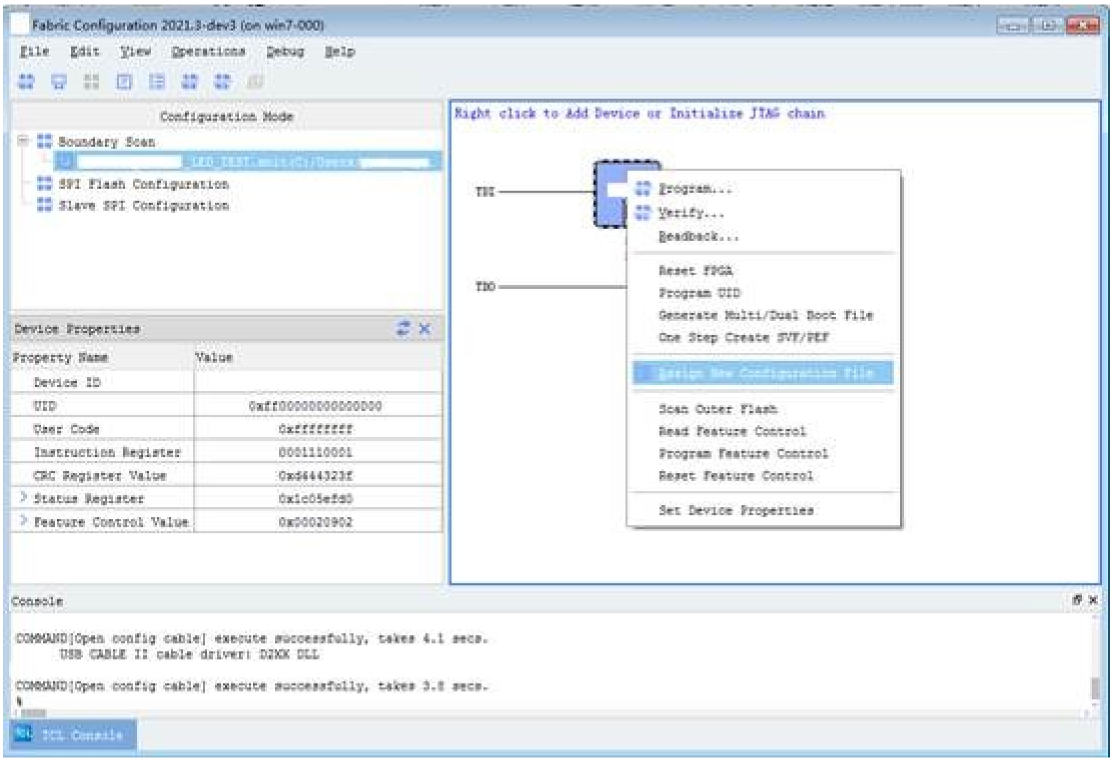


Figure 4 -16 Right-click menu on the device

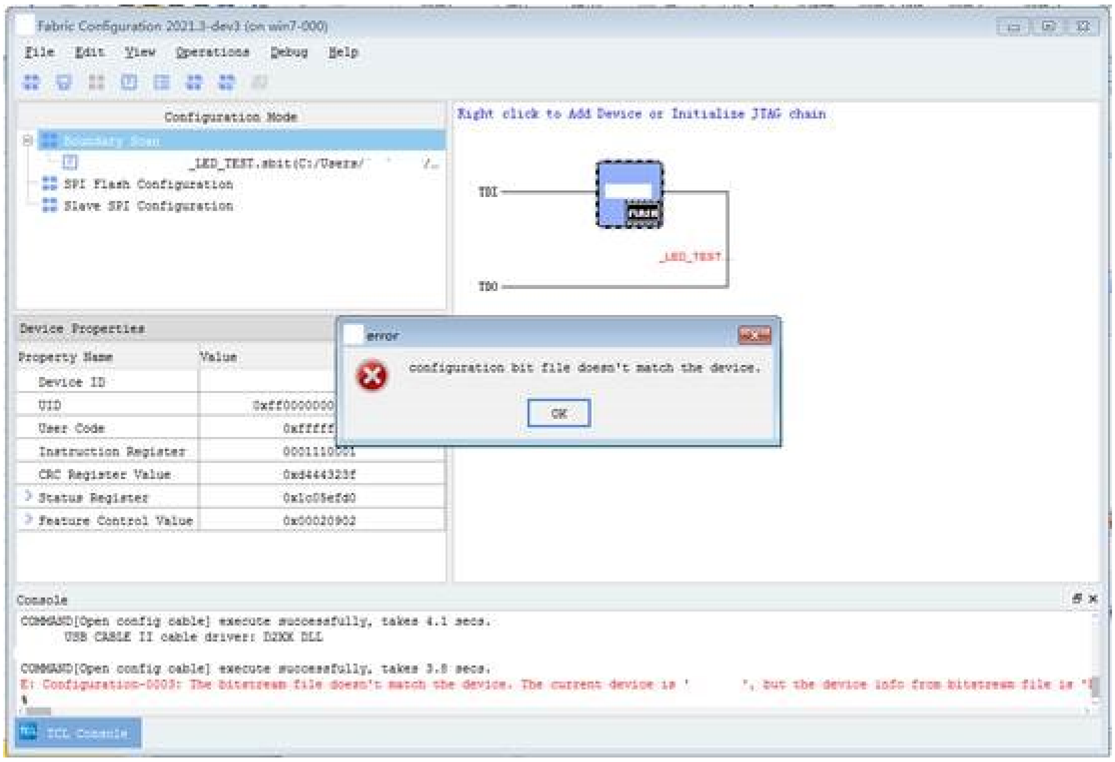


Figure 4 -17 Added mismatching config files

4.3.4 Jtag programming FPGA device

After the logic bitstream file is configured, it can be downloaded to the device. Before performing the download operation, make sure that you have connected to the Jtag Server and configured the bitstream file. After satisfying the downloading conditions, first select the FPGA device on the Jtag chain, then click the right mouse button on the FPGA, a menu will pop up as shown in Figure 4 -16, left click "Program", or left click on the toolbar Click the "Program" button, and Fabric Configuration will download the configuration file to the selected device according to the user's previous preparations (device on the JTAG chain, configuration file).

If the download is successful, as shown in Figure 4-18 ; if it is not successful, there are many possibilities, it may be a hardware or connection problem or the configuration file does not match the device, as shown in Figure 4 -19.

Note: For the operation of the programming key file (nky file), refer to the Jtag programming eFuse register in Section 10 of this chapter.

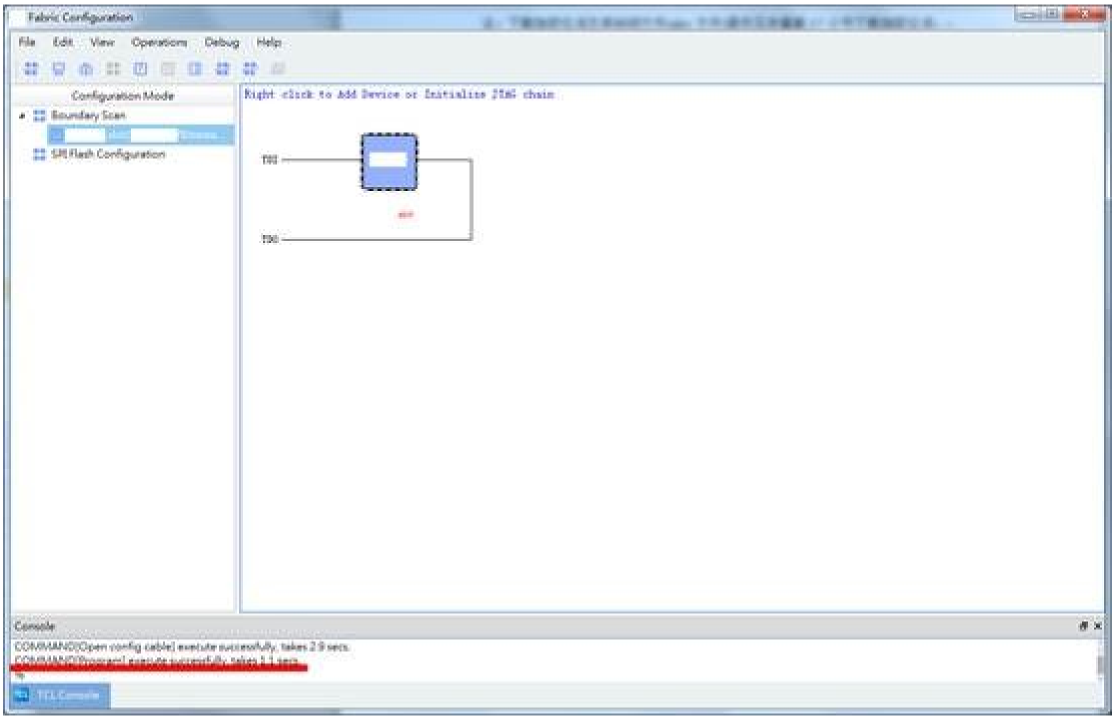


Figure 4-18 Download configuration file successfully

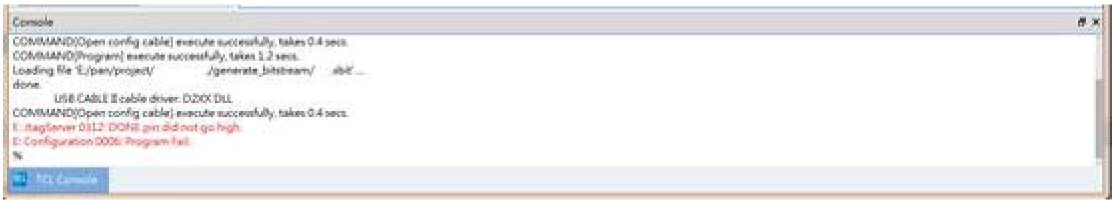


Figure 4-19 download failed

4.3.5 Jtag verification FPGA device

After the operation of programming the bit stream file is completed, in order to ensure that the

downloaded bit stream file has no errors during the download process, it can be verified by a check operation. The verification process is first to read back the data loaded into the device, and compare the read data with the loaded data to verify the correctness of the loaded data.

FPGA verification must provide the smsk file required for verification, During the execution of the verify operation, the user can abort the operation, but the interface will prompt a verification failure message. For FPGA, if the verification fails, the verification error message will be displayed in the information bar and the verification result will be saved in the cfg_verify_result.sbit file (in the log folder of the directory where the application program cdt_cfg.exe is located or in the project directory In the log folder of the directory), and the readback time of the readback file will be named and saved in the rd_year_month_day_hour_minute_second file (in the same directory as cfg_verify_result.sbit). As shown in Figure 4 -20 ; if the verification is successful, it is shown in Figure 4 -21.

Note: The verify operation only compares non- DRM data.

Note: The verify operation is only used when the configured bit stream is uncompressed bit stream and unencrypted bit stream.



Figure 4-20 FPGA verification failed

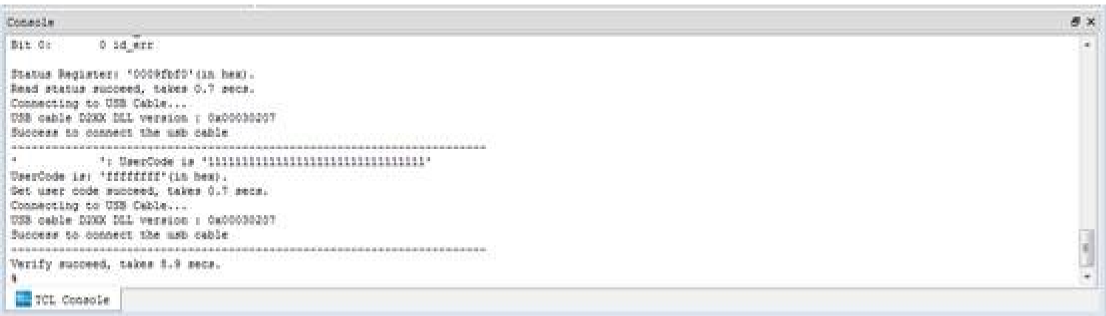


Figure 4 -21 FPGA verification succeeded

4.3.6 Jtag readback FPGA device

When the bit stream file downloaded by the user is uncompressed bit stream and unencrypted bit stream, the user can read back the bit stream data that has been downloaded to the FPGA chip and store it in the file specified by the user.

First select the FPGA device on the Jtag chain, then click the right mouse button on the FPGA, a menu as shown in Figure 4-16 will pop up, left click "Read Back", and the Fabric Configuration dialog

box will pop up for the user to select or create a new one A file, store the read-back data into the selected file, as shown in the figure 4-22.

Note: The Read Back operation is only used when the configured bit stream is uncompressed bit stream and unencrypted bit stream.

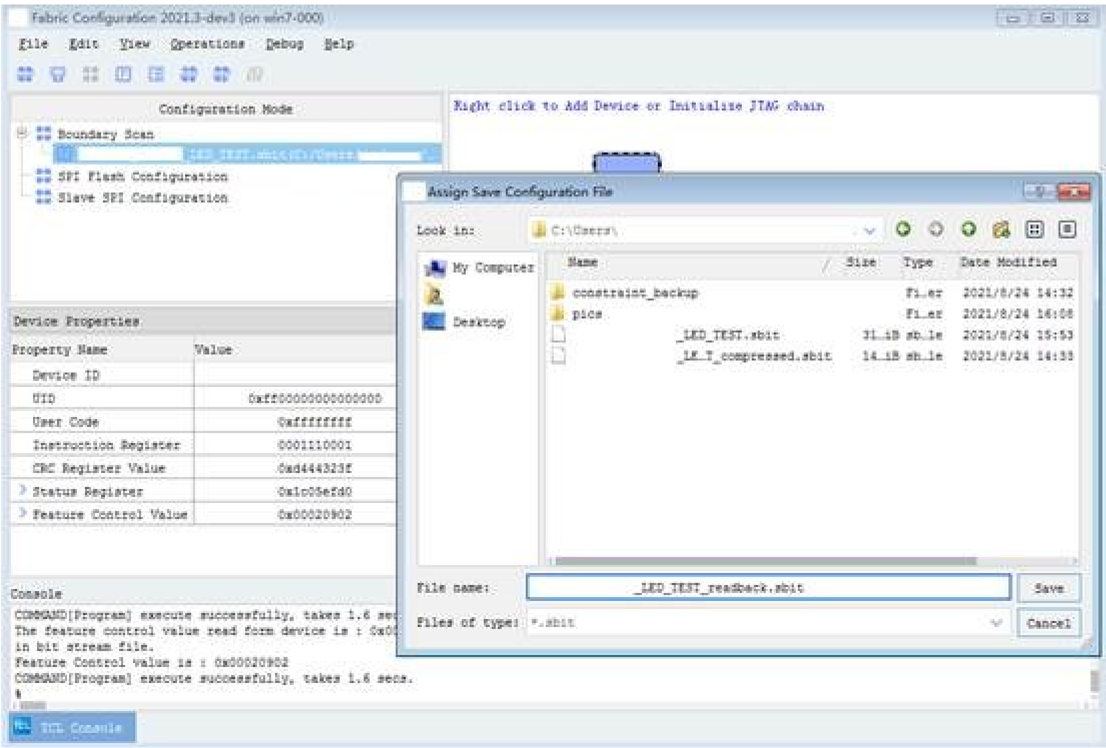


Figure 4 -22 FPGA readback

4.3.7 UID of Jtag programming Compact series

The Compact series supports setting the high eight bits of the device's uniquely coded UID. You can right-click the pop-up menu as shown in Figure 4 -23, click Program UID, enter the target value in the pop-up input box, and click OK to program the high eight bits of the device. Bit UID operation, after the programming is completed, you can check whether the programmed value is correct through the UID in the device property window.

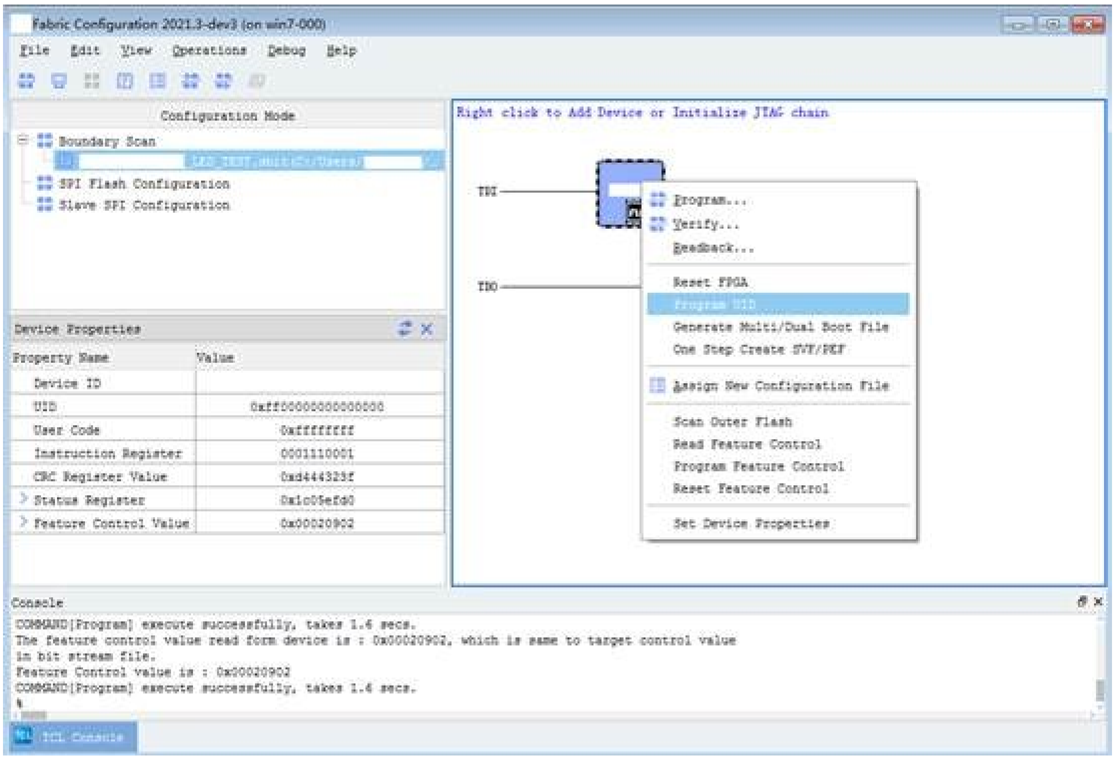


Figure 4-23 Right-click menu

4.3.8 Jtag operates the feature control bits of the Compact series

The Compact series can use or implement some special functions or interfaces by configuring the feature control bits of the device. Users can set feature control bits and perform operations such as programming feature control bits, resetting feature control bits, and reading feature control bits. The user can right-click the menu and click the corresponding menu to operate the feature control bits of the current device, as shown in Figure -4-23.

Note: The programming configuration options of PGC include feature control bits. The feature control bits are used in conjunction with embedded flash and external flash to complete the configuration of PGC.

Note: Other operations embedded in Flash do not need to be reset. Generally speaking, the feature control bit can be written only once.

Note: If the user is not particularly familiar with the feature control bits, he can directly check the default config option to make the chip work normally.

the process of loading FPGA in master-self-loading mode as an example:

1. In the programming configuration option, check the Enable Master Auto Mode In Configuration option, and click Program Feature Control on the interface to enable the master auto mode control.

2. To operate the embedded Flash, configure the bit stream file (the default is the sbit file of the current device), right-click the icon of the embedded Flash and click Program to burn the data.

3. Reset the FPGA by pressing the reset button on the circuit board. At this time, the FPGA loads

the data stream from the embedded flash through the master-self-loading mode.

4.3.9 Jtag operates the embedded Flash of the Compact series

Flash module inside the Compact series device, which supports operations as shown in Figure -4.24. When operating Embed Flash, users can configure the download properties of embedded Flash through the Embed Flash Properties item in the right-click menu- >Set Device Properties-> programming option. For specific setting information, please refer to Section 19 of this chapter, set the properties of the operating device -> set the properties of the operating device of the Compact series.

Note: When downloading the embedded Flash, the bit stream file is the same as that of the main device. If the user wants to configure a bit stream different from the main device for the embedded Flash, he can add the specified sbit file or SFC file when scanning the JTAG chain. can.

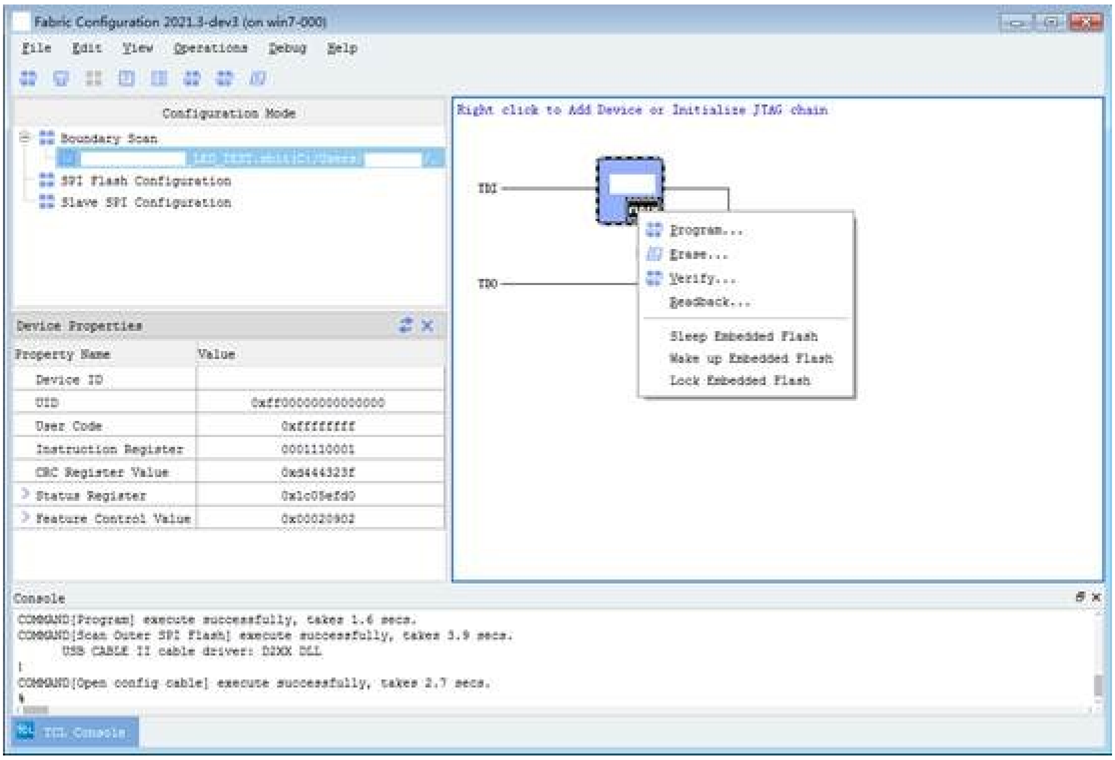


Figure 4-24 operation embedded Flash

1. Program Embedded Flash

Download the bit stream file into the flash chip, the downloaded bit stream is consistent with the bit stream configured by the PGC chip by default; if you want to download the specified bit stream, you can add the specified sbit file or SFC file when scanning the JTAG chain.

When downloading programming, the default is Verify after downloading. If you don't want to perform this operation, you can use the right-click menu- >Set Device Properties->Embed Flash Properties to remove the corresponding check option.

If you need to start downloading the bit stream from a specified page, it can be realized by setting Embed Flash Start Page Index. Assuming the setting value is 20, the bit stream will be downloaded from the 20th page of Embed Flash.

2, Erase Embedded Flash

erasing flash data, page erasing and chip erasing; the default is no erasing, and you can use the right-click menu-> Set Device Properties->Embed Flash Properties->Flash Entire Flash to perform the mode choose.

If it is necessary to start erasing from a specified page, it can be realized by setting Embed Flash Start Page Index. Assuming that the setting value is 20, the Embed Flash will be erased from the 20th page of the Embed Flash. If the erasing method is selected as the whole chip erasing, this setting parameter will be invalid.

Note: All memory cells of Compact series embedded Flash are set to 0 after being erased, and all memory cells of SPI Flash and external Flash are set to 1 after being erased.

3. Verify Embedded Flash

the Flash device according to the given file size, and then compare the read data with the loaded data to verify the loaded data. correctness of the input data. If the comparison fails, the FlashReadBack_year_month_day_hour_minute_second.sfc file will be generated (the file will be stored in the same directory as the cdt_cfg.exe software).

4. Read Back Embedded Flash

This function is used to read the data in Flash and store the read data in the file specified by the user. If the downloaded bit stream starts to download the bit stream from a specified page, then when performing the readback operation, it is also necessary to set the readback bitstream from a specified page. It can be realized by setting Embed Flash Start Page Index.

This function also supports specifying the size of the read back content, which can be set through the right-click menu- >Set Device Properties->Embed Flash Properties->Embed Flash Read Back Size. Suppose the Embed Flash Start Page Index is set to 20, If the setting value of Embed Flash Read Back Size is 1000, the read back starts from the 20th page of Embed Flash, and the read back content size is 1000 bytes.

5. Sleep Embedded Flash

The sleep operation is to put the embedded Flash in a sleep state and not process any operations on the flash.

6. Wake up Embedded Flash

The wake-up operation is to make the embedded Flash in the working state and process the corresponding operations.

7. Lock Embedded Flash

to lock the relevant area according to the starting address of the programming configuration and the size of the flash file, and cannot be read or written. Unlocking the state requires erasing the entire flash. The default locking range is the entire slice. If you only need to go from the beginning to a

specified page, it can be realized by setting Embed Flash Tail Page Index. Assuming the setting value is 20, the first to twentieth pages of Embed Flash will be locked.

4.3.10 Jtag programming eFuse register

1. Programming key file

The nky file is a configuration file for encrypting and decrypting the bit stream file, and the nky file contains a key for decrypting the bit stream file. There are two ways to configure the key:

①, inner key, the key configuration information can be written to the device multiple times, but when the device is powered off, the configured key information will disappear and need to be written again.

②, efuse key, can only write the key configuration information to the device once, after which the information cannot be changed and the information will be permanently stored in the device. When the device is powered off, the configured key information will not disappear. When the nky file of the inner key is downloaded to the device, the subsequent bit stream file must be a bit stream file encrypted by the key in the nky file, so that the download can be successful in the device; otherwise, the unencrypted bit stream file The file will not download successfully. The nky of efuse key does not have this limitation.

The specific operation is as follows:

1. Select " Program eFuse Registers " in the pop-up menu after right-clicking, and the following figure will pop up 4 -25, select the Program Key File operation type, and click Next to enter the window for selecting the key file.

2. In the Configure Key File window, the user selects the nky file to be programmed in the file input box, and the software will analyze whether the key file matches the currently selected device, and if it matches, it will print out the key information, such as Figure 4 -of 26 shown. After selecting the key file, click Next.

3. After selecting the secret key file, the selected operation type and secret key file information will be printed out on the summary page for the user to confirm, as shown in the figure 4 -of 27 shown. If you want to modify the key file, you only need to go back and select a new file. After confirming that the information is correct, click Finish to start programming. If the programmed key is a permanent key, since the permanent key can only be programmed once, the user needs to confirm again, as shown in the figure below 4 -of 28 shown.

Note: After the permanent key is programmed, the key file can no longer be modified, please operate with caution!

4. After downloading the key file nky, you can configure the corresponding encrypted bit stream. Select the target encrypted bit stream and download it. If the selected encrypted bit stream is the bit stream encrypted by the downloaded key, the download can be successful, otherwise, the download will fail.

Note: The encrypted bit stream cannot be read back and verified.

Note: After configuring the key file, only the corresponding encrypted bit stream can be configured, but other non-encrypted bit streams can also be configured.

In addition, for the devices of Logos2 and Titan2 series, there is also the function of programming BBRAM (power supply key register), and the user can burn the key file (*.nky) into the power supply RAM. Just select BBRAM on the Set Key Location page, as shown in the figure 4 -29.

You can only choose to use one of the two registers eFUSE and BBRAM for programming the key file, and you cannot use both at the same time, otherwise errors will easily occur.

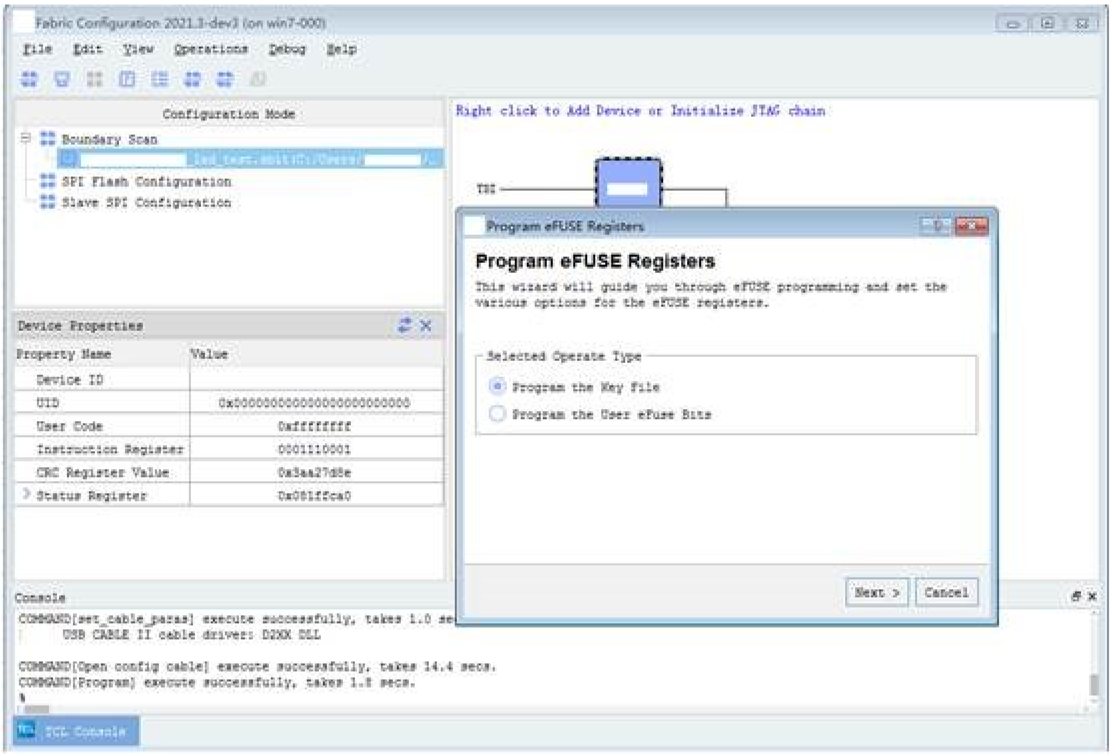


Figure 4-25 select operation eFuse type

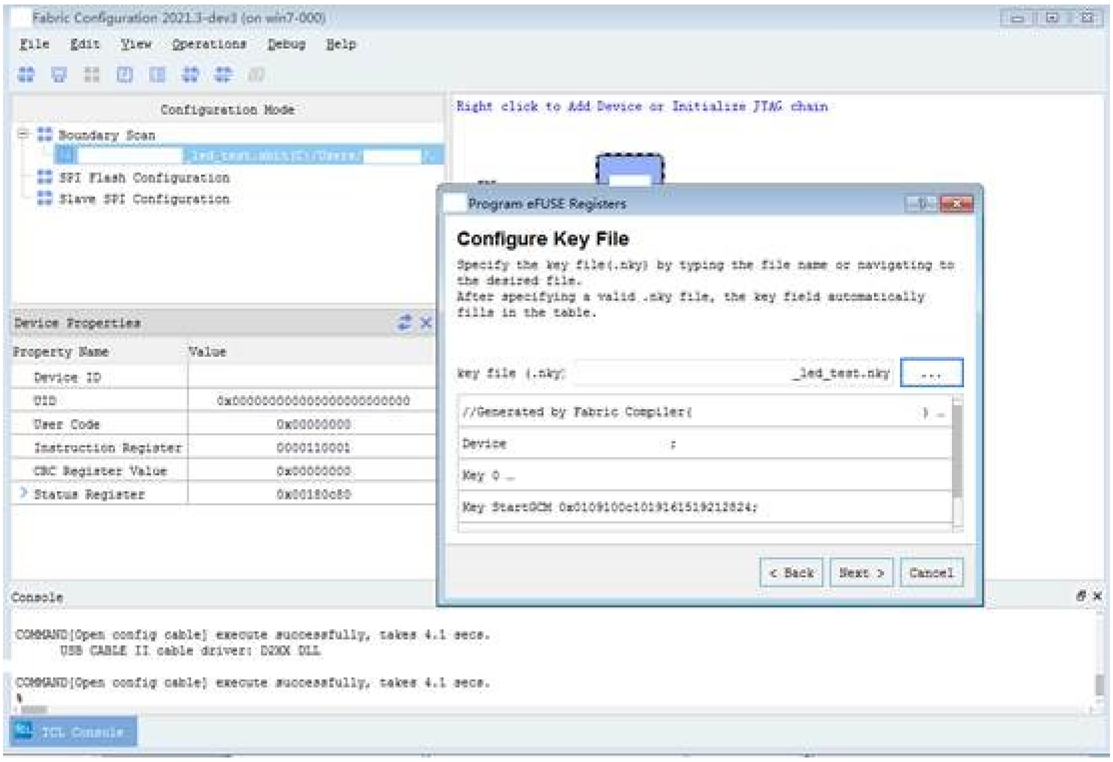


Figure 4-26 Configure the key file

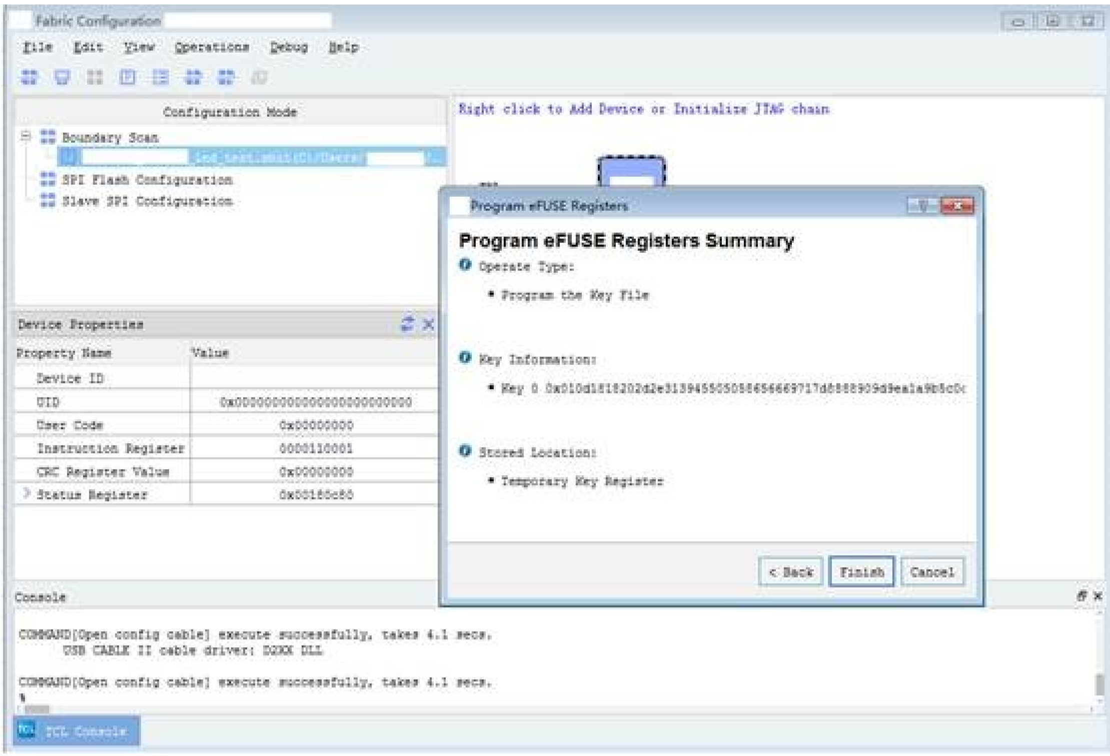


Figure 4-27 Configure key information

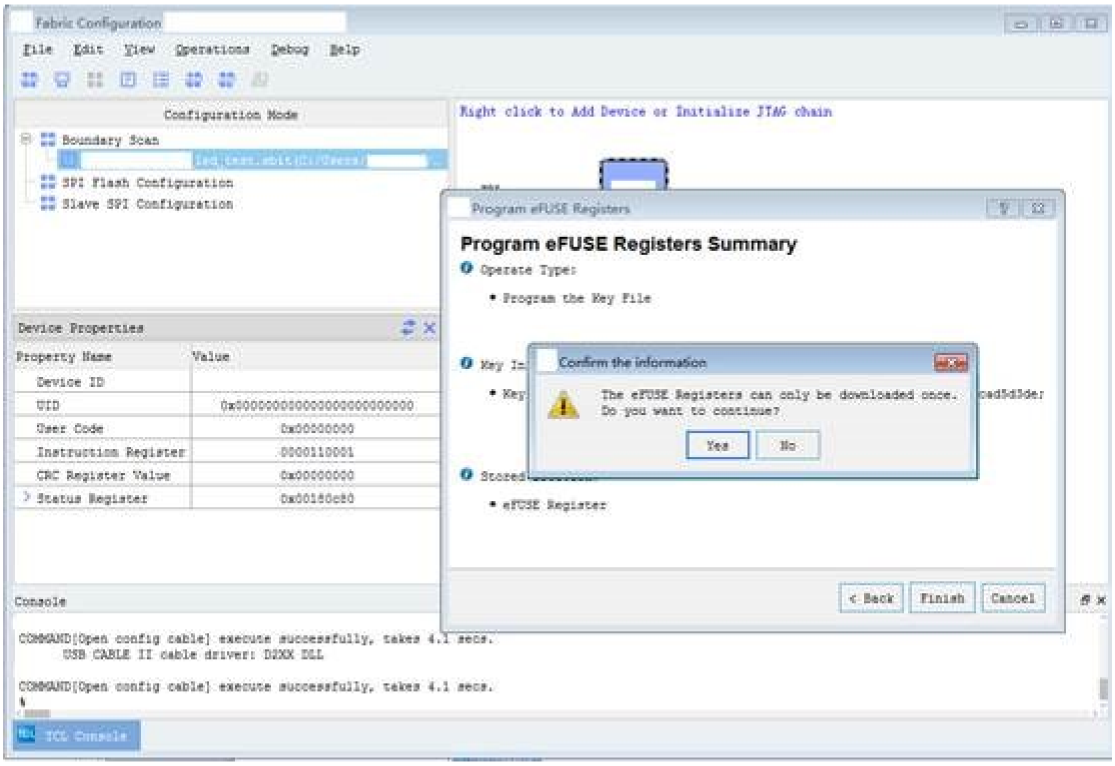


Figure 4-28 Warnings when configuring permanent keys

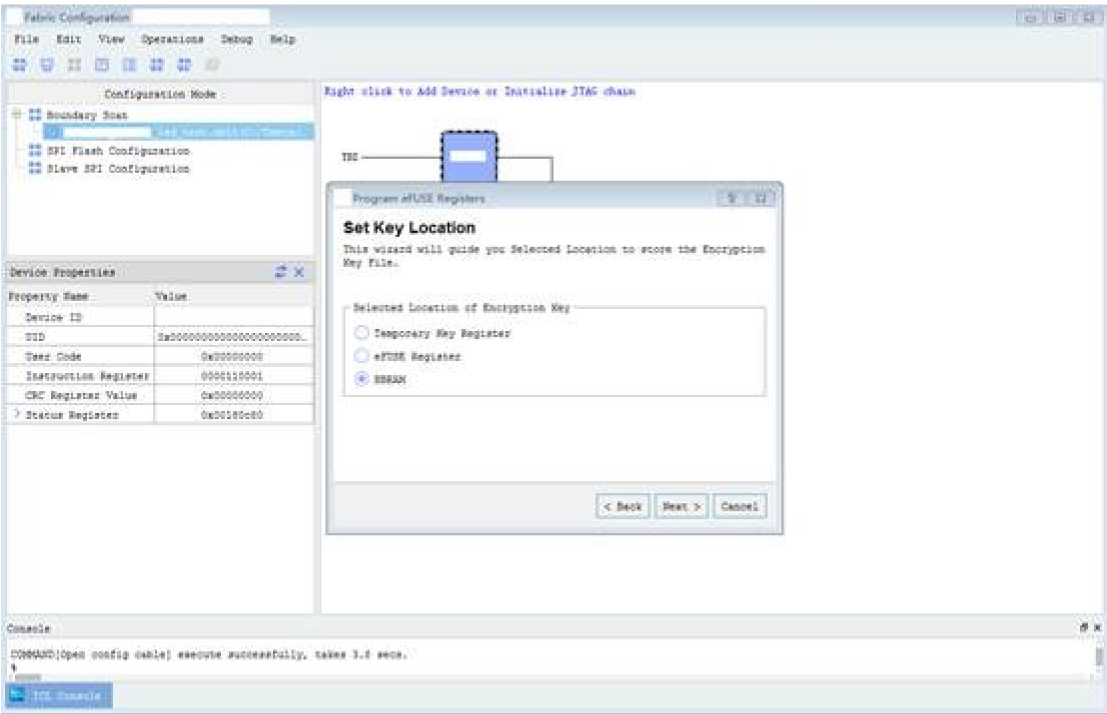


Figure 4-29 Select the key file to write to BBRAM

2. Program 32 -bit User eFuse

Efuse is a one-time programmable non-volatile memory. The 32 -bit Efuse (used to store the key) has an unprogrammed fuse value of 0 and a programmed fuse value of 1. The Config Efuse Value operation is mainly for the Titan series and Logos series. The Config Efuse Value can only be written to the target device once, so it should be operated with caution.

When the device is ready, you can configure the Efuse memory of the device, the specific operation is as follows: in the figure 4 In -25, select the "Program User eFuse Bits" operation type, click Next, and a pop-up will pop up as shown in the figure 4 -30 input box, the user can input 32 -bit efuse binary information. After the input is complete, click Next to the summary page and click Finish to program.

Since writing 32 -bit efuse binary information into the Efuse memory device is a one-time operation, after writing, the information in the Efuse memory cannot be changed, so it needs to be operated with caution. After clicking Finish, a confirmation box will pop up, and you can program the eFuse only after confirming again.

If the writing of the Efuse configuration is successful, the corresponding writing success message will be displayed in the information column, and the 32bits efuse value can be checked in eFuse->User Bits in the Device Properties window. If unsuccessful, there are a number of possibilities, it could be a hardware or connectivity issue or a configuration file mismatch with the device.

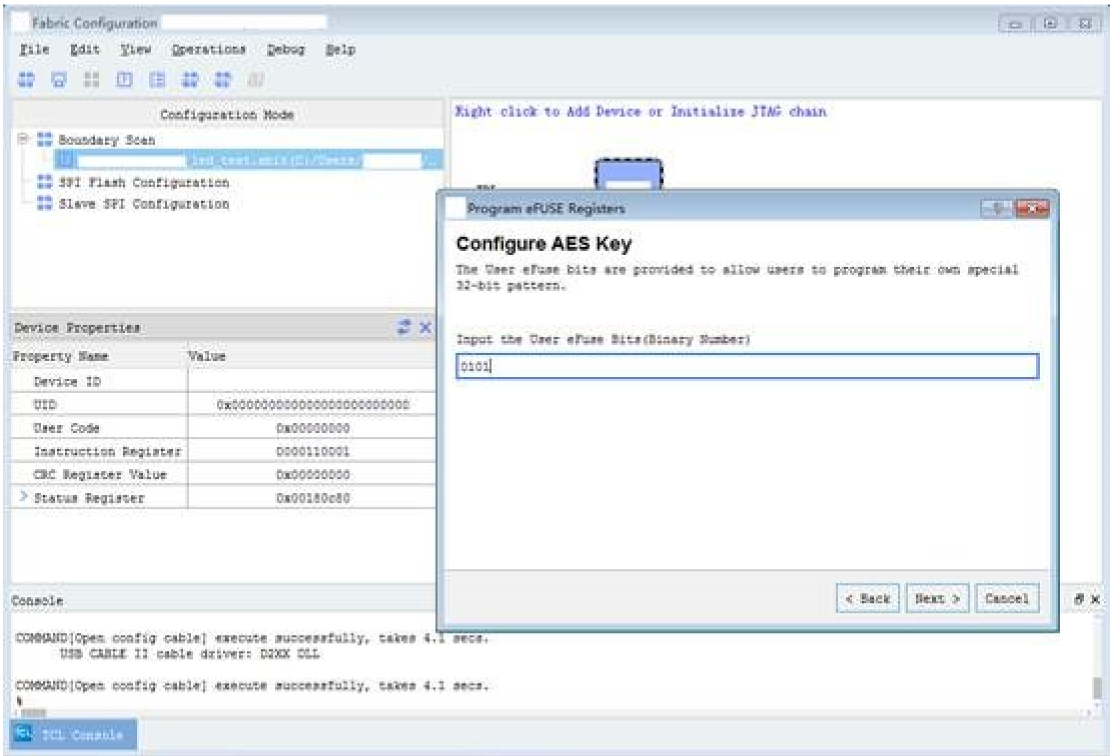


Figure 4-30 32bits User eFuse configuration interface

4.3.11 Jtag soft reset FPGA

Fabric Configuration supports the reset operation by sending commands through software. The operation is as follows: Right-click on the selected device, a pop-up will pop up 4 -In the menu shown

in 16, left-click "Reset FPGA", and Fabric Configuration will send a reset command to reset it.

For devices of the Compact series, some operations, such as modifying feature control bits, will take effect after reset. If you do not want to reset by power down, you can use this software reset operation to achieve.

4.3.12 Jtag scans Flash devices

Logos and Compact series support direct programming of external Flash. You can right-click the pop-up menu and click Scan Outer Flash to scan the external Flash device. Take PGL50H as an example, as shown in the figure below 4 of -31.

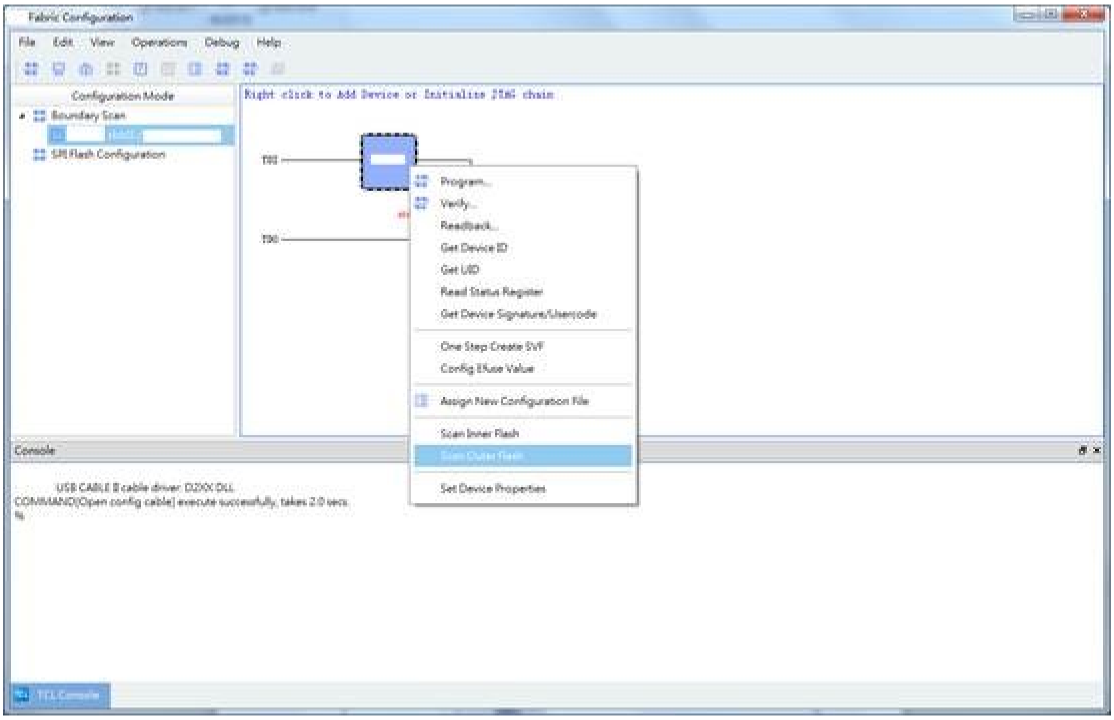


Figure 4-31 PGL50H operation menu

After successfully scanning the external Flash, you can use the JTAG interface to perform Erase, Program, and Verify operations on the Flash device, as shown in the figure below 4 -32 shown. The download property configuration of Flash is the same as that of SPI Flash.

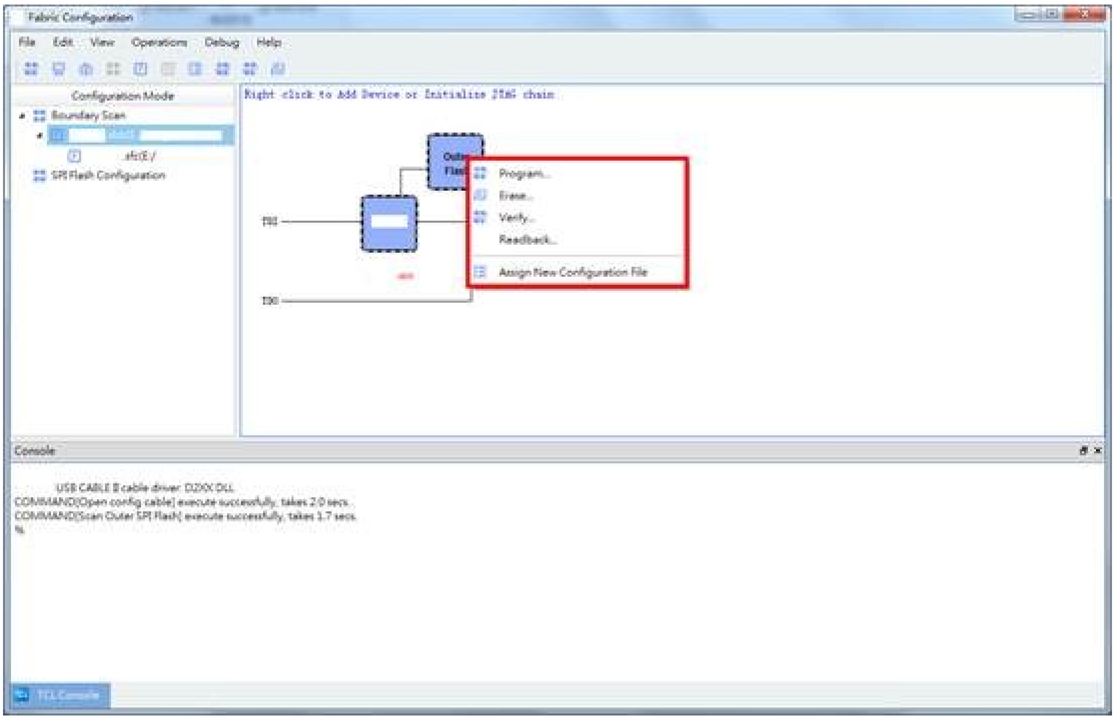


Figure 4-32 JTAG operation Flash device

4.3.13 SPI scan Flash device

Configuration Mode window on the left side of the main interface, select the SPI Flash Configuration mode, then right-click the mouse in the work area, left-click Scan Device, or left-click Scan Device in the File menu or toolbar, all will be Execute the operation of scanning SPI Device.

If the SPI Device scan is successful, all scanned devices will be listed in the work area, the device model will be displayed in the information column, and a dialog box will pop up to enable the user to add a configuration file for the device as shown in the figure As shown in 4 -33, the schematic diagram of the successful scanning of SPI Device is shown in the figure 4 -of 34 shown.

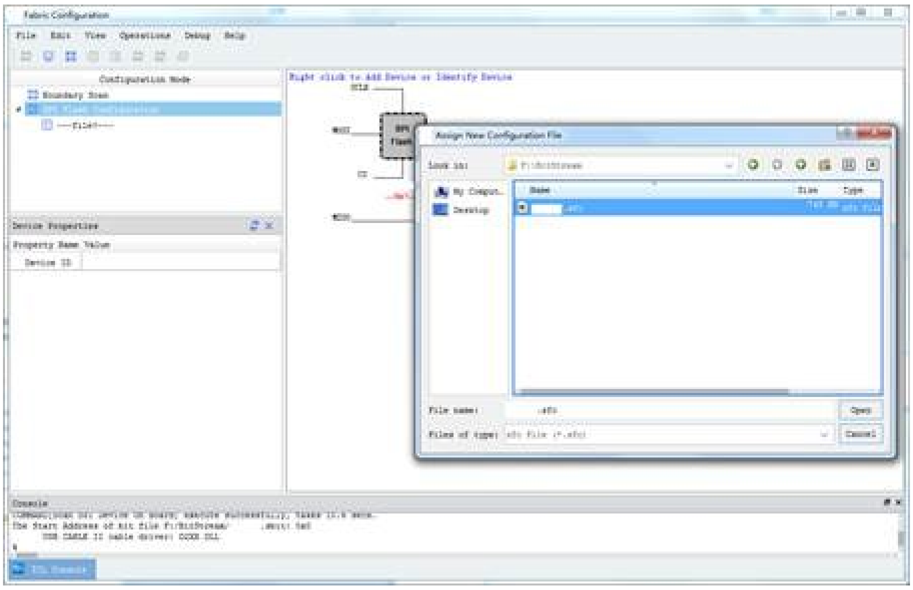


Figure 4 -33 Add.sfc file

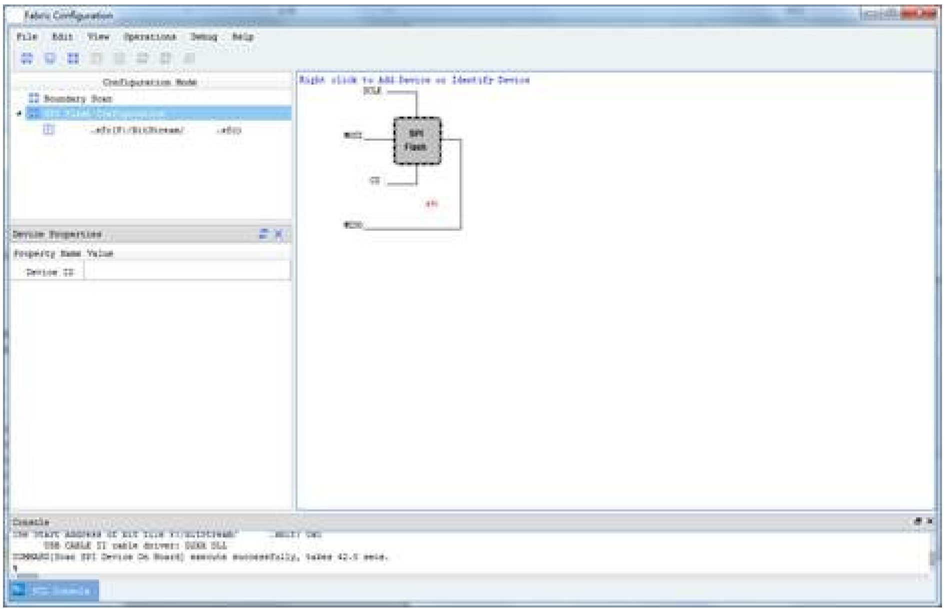


Figure 4 -34 Scanning SPI Device successfully

4.3.14 SPI Erase Flash Device

as shown in the Figure As shown in 4 -35, the Erase function can erase the content of the SPI device. The specific operation is as follows: Right-click and select "Erase", or left-click "Erase" in the toolbar, and Fabric Configuration will perform an erase operation on the SPI Flash device. The user can select the erasing method and the starting block to be erased through the right-click menu -> Set Device Properties. Currently, Configuration supports block erase, full chip erase, sector erase, and page erase. The Auto method automatically selects the erase method according to the size of the selected bit stream and the capacity of the SPI Flash. If the user chooses to erase the entire chip and also sets the starting block of the erasing, then the starting block address of this setting will be ignored.

For the specific description of the erasing properties of Flash, please refer to Section 19, Setting Device Properties, Erasing Properties of SPI Flash.

Note: The erased memory cells of SPI Flash and external Flash are all set to 1, and the erased memory cells of Compact series embedded Flash are all set to 0.

4.3.15 SPI programming Flash device

as shown in the Figure 4 -35, the Program function loads the bitstream configuration information into the SPI device. The specific operation is as follows: Right-click and select "Program", or left-click "Program" in the toolbar, and Fabric Configuration will perform a download operation on the SPI Flash device. Users can select the download method and the starting block for download through the right-click menu -> Set Device Properties. Currently, Configuration is divided into three modes : Typical Program (the waiting time is in the middle), Fast Program (the waiting time is the shortest) and Slow Program (the waiting time is the longest) according to the waiting time for each page of Flash

download. The Slow Program can guarantee the success of the Flash download, and the others can also be successful when the frequency of the Cable setting is not very high.

In general, if SPI Flash programming fails, you should consider reducing the frequency of Cable and choose a download mode with a longer waiting time.

4.3.16 SPI verification Flash device

as shown in the Figure As shown in 4 -35, the process of SPI verification is first to read back the data loaded into the SPI Flash device, and compare the read data with the loaded data to verify the correctness of the loaded data.

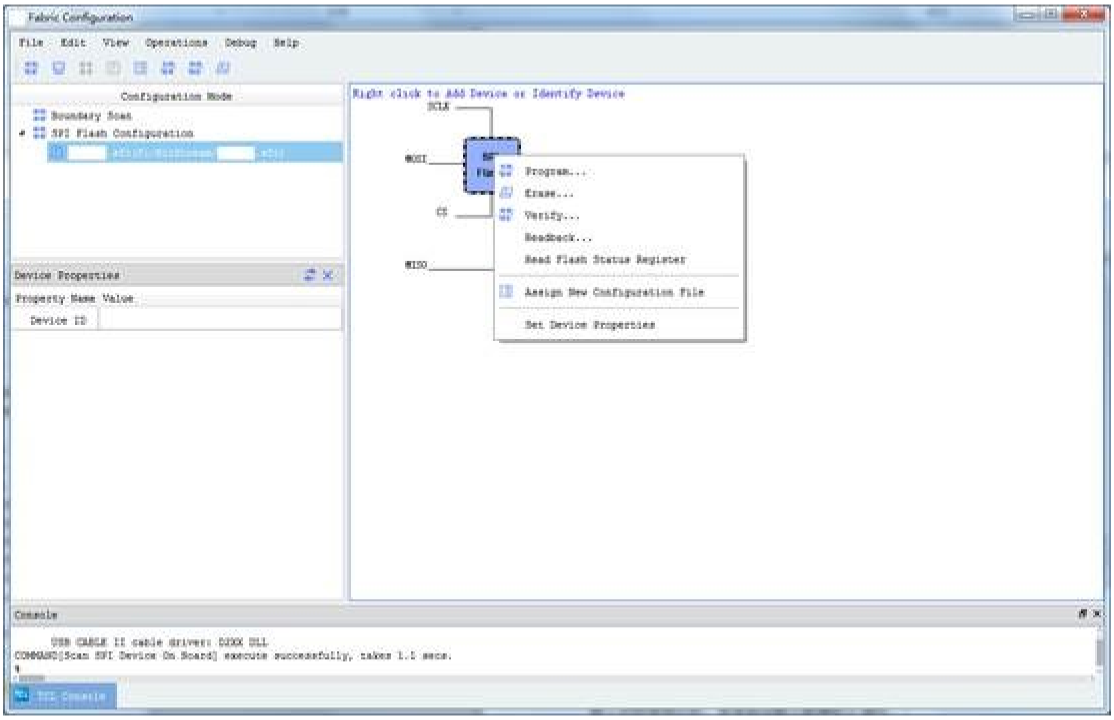


Figure 4-35 Operations supported by SPI Flash

4.3.17 SPI readback Flash device

as shown in the Figure As shown in 4 -36, the software supports reading back the data in SPI Flash and storing it into the file specified by the user. Right-click on the SPI Flash device, then left-click "Read Back", and Fabric Configuration will pop up a dialog box to allow the user to select or create a file for reading back data storage.

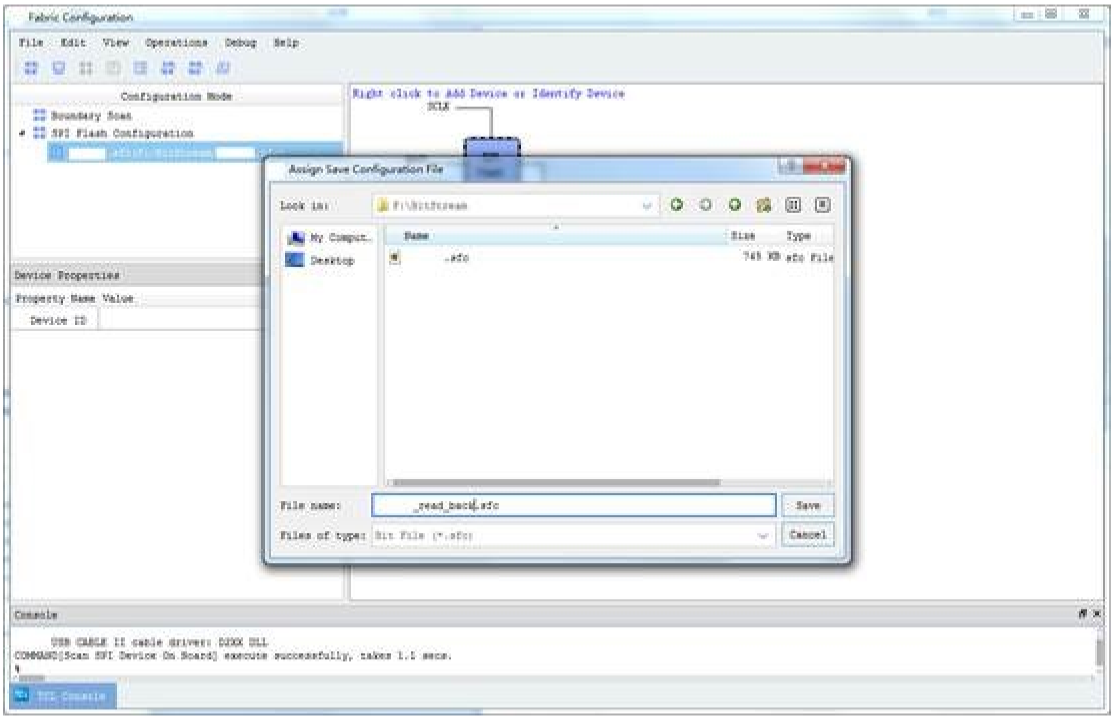


Figure 4-36 Read back SPI Flash data storage file

4.3.18 SPI read and write Flash register

As shown in Figure -4.35, users can read the SPI Flash register to know which mode the current Flash is in. Since the register values of each manufacturer are inconsistent, it is necessary to refer to the corresponding manual to understand the meaning expressed by each bit of the register..

The specific operation is as follows: right-click and select "Read Flash Status Register", or left-click "Read Flash Status Register" in the toolbar, and Fabric Configuration will perform the read register operation on the SPI Flash device. If the reading is successful, the register value will be printed in the information display column, as shown in Figure 4-37.

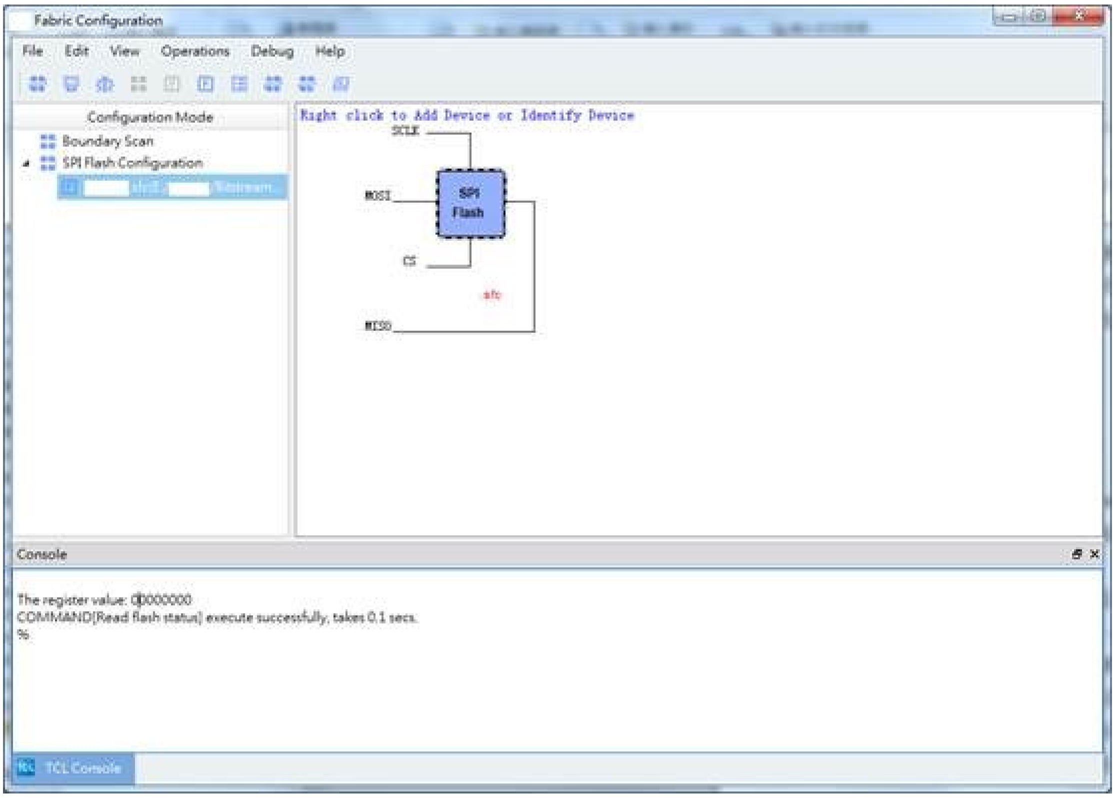


Figure 4 -37 SPI Flash status register

Writing the register value can only be realized through the Tcl command cfg_flash_write_register. For details, please refer to the SPI Flash section in the fourth chapter of Tcl command introduction.

4.3.19 Set operation device properties

the Set Device Properties option under the right-click menu to set the Program property, Read Back property and Flash -specific Erase property of FPGA and Flash- related devices. The device property setting of FPGA devices supports Titan2 series, Logos/Logos2 series and Compact series, Flash Only supports external Flash (Jtag Flash) and SPI Flash.

1. Set the operating device properties of the Titan2 series

As shown in Table 1, the programming options specifically supported by the Titan2 series are as follows:

Table 1 Programming properties of Titan2 series

Attributes	meaning
Enable WakeDownReconfigure	Indicates whether power-off reconfiguration is allowed.
Enable ISC Configure	Indicates whether to use ISC for programming.
Enable Wake Down	Indicates whether shutdown readback is allowed.
Enable ISC Read Back	Indicates whether to use ISC for readback.

2. Set the operating device properties of the Logos/Logos2 series

The configuration options supported by the Logos/Logos2 series are shown in Table 2 :

Table 2 Programmatic properties of the Logos family

Attributes	meaning
Enable WakeDownReconfigure	Indicates whether power-off reconfiguration is allowed.
Enable ISC Configure	Indicates whether to use ISC for programming.
Enable Wake Down	Indicates whether shutdown readback is allowed.
Enable ISC Read Back	Indicates whether to use ISC for readback.

3. Set the operation device properties of the Compact series

The configuration options supported by the Compact series are shown in Table 3 :

Table 3 Programming properties of the Compact series

Attributes	meaning
Enable WakeDownReconfigure	Indicates whether power-off reconfiguration is allowed.
Enable ISC Configure	Indicates whether to use ISC for programming.
Enable High ZDis Configure	Indicates whether to use offline programming
Enable Wake Down Read Back	Indicates whether shutdown readback is allowed.
Enable ISC Read Back	Indicates whether to use ISC for readback.
Enable High ZDis Read Back	Indicates whether to perform offline readback
Enable Erase before Program EFlash	EFlash before downloading the bitstream
Enable Verify after Program EFlash	EFlash after downloading the bitstream
Enable Erase Entire Embed Flash	Indicates whether to erase the entire chip when erasing Eflash
Enable Reset FPGA after Erase Embed EFlash	whether to enable reset after erasing EFlash
Enable Reset FPGA after Program Embed EFlash	whether to enable reset after programming EFlash
Enable Reset FPGA after Program Feature Control	Indicates whether reset is enabled after programming the feature control bit
Enable Program Feature Control after Program EFlash	Indicates whether to enable the programming feature control bit after downloading the bitstream
Enable Program Feature Control before Program CRAM	Indicates whether to enable the programming feature control bit before programming the CRAM
Embed Flash Start Page Index	Operate the start address page of Embed Flash, which is used to set the start address page of programming, erasing and readback.
Embed Flash Tail Page Index	Operate the tail address page of Embed Flash, which is used to lock the contents of Embed flash from the beginning to the specified page.
Embed Flash Read Back Size	Specifies the readback content size starting from the start address page of the Embed Flash operation, in Byte.

Among them, Enable ISC Configure and Enable HighZDis Configure are mutually exclusive, and Enable ISC Read Back and Enable HighZDis Read Back are mutually exclusive, that is, they cannot be selected at the same time.

Note : For the operation instructions of the Compact series feature control bits, see Section 8 of this chapter, Operation Feature Control Bits

Note : For the programming operation of the Compact series embedded Flash, refer to Section 9

of this chapter, Operating the Embedded Flash.

4. Set the operating device properties of Flash

Flash (Jtag Flash) and SPI Flash, users can configure the corresponding properties on this page. The specific information is as follows.

1. Flash Device Properties

Flash Start Address : Display the start address of Flash programming download or erase or readback.

Flash Start Block Index(Block Size: 64KB) : The starting block index of Flash programming download or erase or readback, the setting unit is 64KB. (Example: If the value set by Flash Start Block Index is 1, it means that the starting address of Flash programming download is 0x10000).

Flash Start Sector Index(Block Size: 4KB) : The index of the starting sector set when the Flash sector is erased, and the setting unit is 4KB. (Example: If the value set by Flash Start Block Index is 1, and the value set by Flash Start Sector Index is 2, it means that the starting address of Flash erasing is 0x12000).

Flash Read Back Size(Unit: Byte) : Set the read back Flash size.

2. Flash Program Properties

Flash Program Mode : SPI Flash programming download download speed level selection, respectively Typic, Fast, Slow three modes, the speed decreases in turn, if the user is unstable when downloading Flash, the download speed can be reduced.

The Flash Erase before Program option indicates that the content in the flash will be automatically erased before programming. If the erase is successful, then the programming download will be performed.

The Flash Verify after Program option indicates that after the programming is successful, it will automatically verify the consistency between the content in the flash and the content of the sfc file.

3. Flash Erase Properties

Flash Erase Mode contains five modes: Sector Erase, Block Erase, Chip Erase, Page Erase, Auto.

Sector Erase : means erase by sector.

Block Erase : means erase by block.

Chip Erase : Indicates that all Flash devices are erased.

Page Erase : means to erase the page of the Flash device.

Auto : Indicates that the program automatically selects the appropriate Block or Chip is erased to ensure that the erasing time reaches the minimum.

4.3.20 FPGA bit stream file conversion of multiple format data streams

Click the "Convert File" option in the Operations menu and a dialog box will pop up as shown in the figure As shown in 4 -38, its main function is to convert FPGA bit stream files into files in various formats, such as for Flash configuration, daisy-chain devices, and the realization of some FPGA device functions. Specifically, there are four forms of file conversion: ① Generate Flash Programming File ; ② Generate Daisy Chain File ; ③ Generate Multi Revision File ; ④ Generate Chain Svf File.

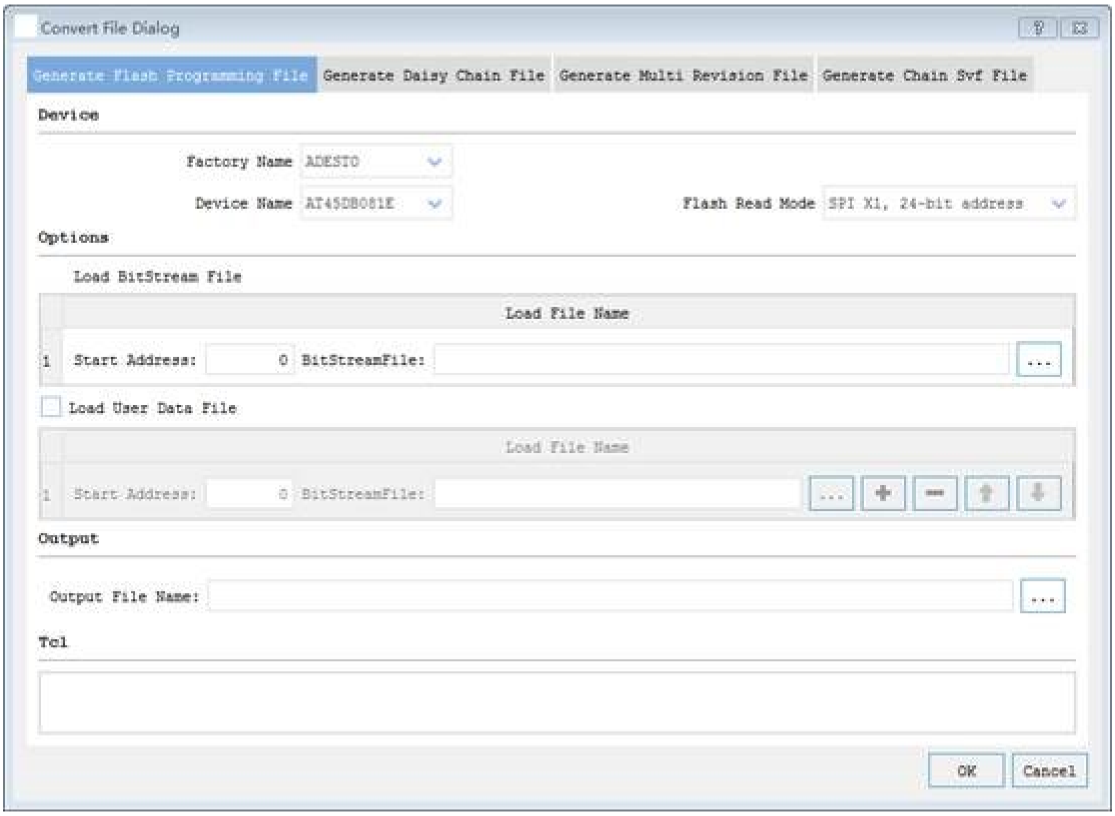


Figure 4 -38 Convert File Dialog

1. Generate Flash Programming File

This function is mainly used to convert and generate SFC files that can configure Flash. by figure As shown in 4 -39, in the Generate Flash Programming File window, first select the corresponding Flash device and its read/write mode. Then select the bit stream file to be converted. If the converted bit stream file is Logos series, you can also select the edge of data sampling and whether to generate SPI X8 mode, as shown in the figure 4 -of 38 shown.

If the user wants to configure the start position of the bit stream in Flash, he can modify the start address of the bit stream by entering the address in Start Address, as shown in the figure As shown in 4 -39 Generate Flash Programming File, if the address entered by the user is 0x24, the software will add 0x24 FFs at the beginning of the bit stream during conversion. In other words, the starting address of the bit stream is relative to the beginning of the Flash chip offset by 0x24 bytes. If the user does not need to add user data, he can directly click OK to convert.

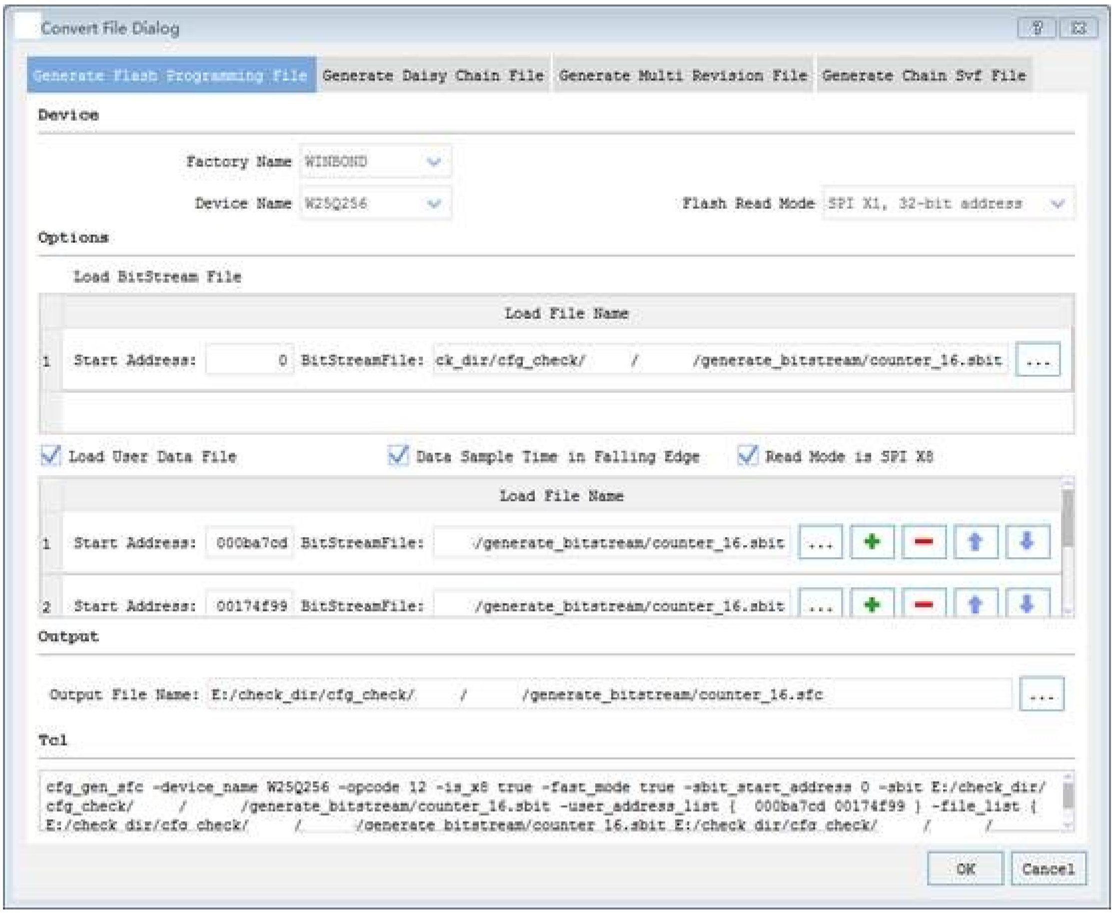


Figure 4 -39 Generate Flash Programming File

If the user needs to mask the time information in the bit stream, the mask sbit time info option can be checked.

If the user needs to add their own data behind the bit stream, they can check Load User Data at this time, then add their own user data in the Load File Name table, and adjust and delete the user data according to the provided buttons, such as As shown in Figure 4-39. Generally speaking, the start address of user data follows the bit stream by default, and the software will automatically calculate the start address. If the user wants to modify the start address of the user data, it must be noted that the start address value must be greater than the sum of the size of the bit stream file and the size offset by the start address of the bit stream file. If the input address is incorrect, the software will automatically correct the address to the default value.

Note: The conditions for adding user data are: there must be bit stream data.

After adjusting the position of the bit stream file and the user data file, click the OK button to generate the SFC file. If too many user data files are added, it may take a long time to generate and you need to wait for a while.

Currently supported flash devices and their matching read/write modes are shown in Appendix IV.

2. Generate Daisy Chain File

This function is mainly to generate cascade data stream download files. As shown in Figure 4-40, in the Generate Daisy Chain File window, add the sbit file that generates the cascaded data flow, and

the serial number in front of the file is the location of the actual FPGA chain. The user can adjust the position through the buttons. It must be noted that the device corresponding to each data stream that generates the cascaded data stream must correspond to the device position downloaded to the actual FPGA chain.

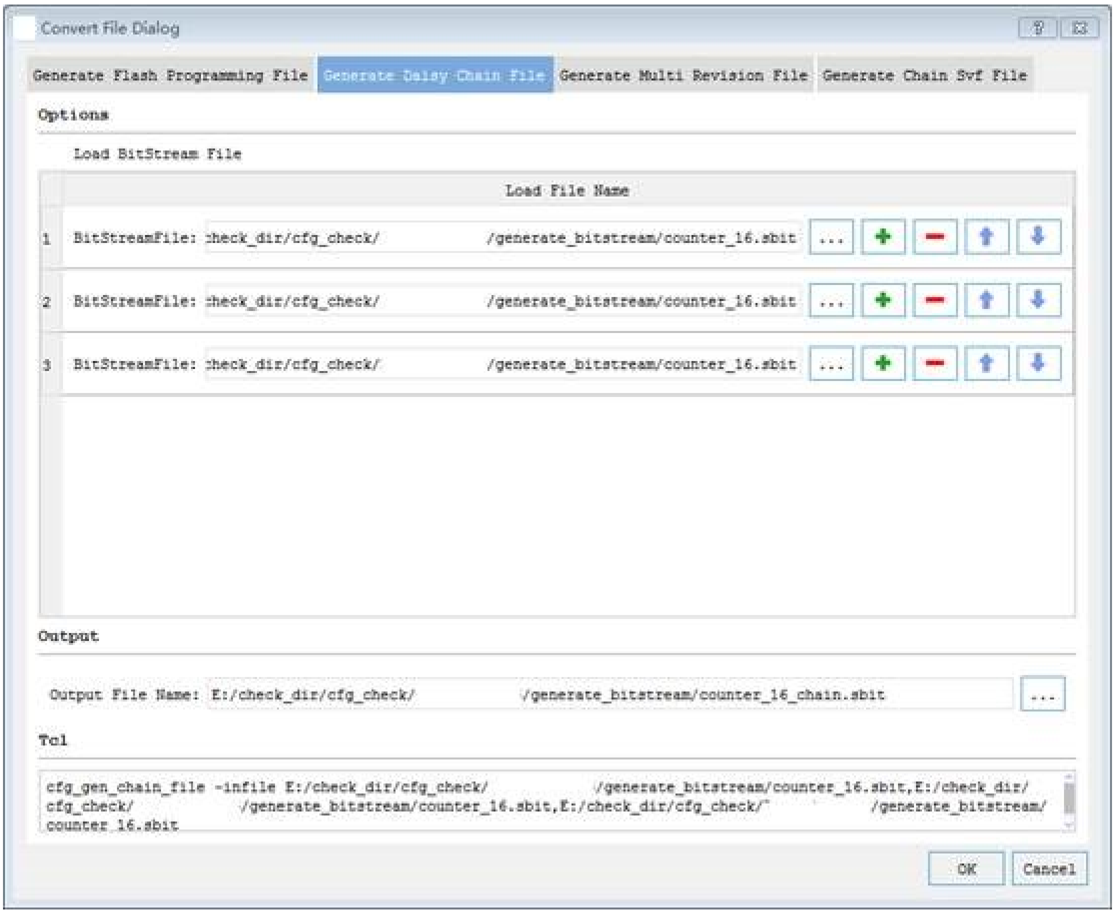


Figure 4-40 Generate Daisy Chain File

After selecting the bit stream file and adjusting the position of the file, click the OK button to generate the cascaded data stream. If too many bit stream files are added, it will take a long time to generate and you need to wait for a while.

3. Generate Multi Revision File

As shown in Figure 4-41, its main function is to generate Flash download files of the golden data stream and combined data stream. Users can select the corresponding device in Device Family Type and select the bit stream format to be generated in the Data Stream Type selection box. Currently supported formats are:

①. SPI Upgrade Data Stream : SPI (Serial Peripheral Interface) remote upgrade data stream can only be generated correctly when there are at least two application data streams.

②, BPI Upgrade Data Stream : BPI (Byte-wide Peripheral Interface) data stream can only be generated correctly when there are at least two application data streams.

③, Multi Boot Data Stream : multi-boot data stream, only Logos, Logos2, Titan2 series support. Properly generated when there are at least two application streams.

④, Multi Function Data Stream : Multi-functional data stream, only supported by Logos series. Properly generated when at least two application streams exist.

⑤、 Master Auto And Dual Boot Data Stream. Main self-loading dual boot data flow, only supported by Compact series. Generation is possible if and only if there are two application data streams.

Note: Combining data streams supports up to 4 data streams for combination.

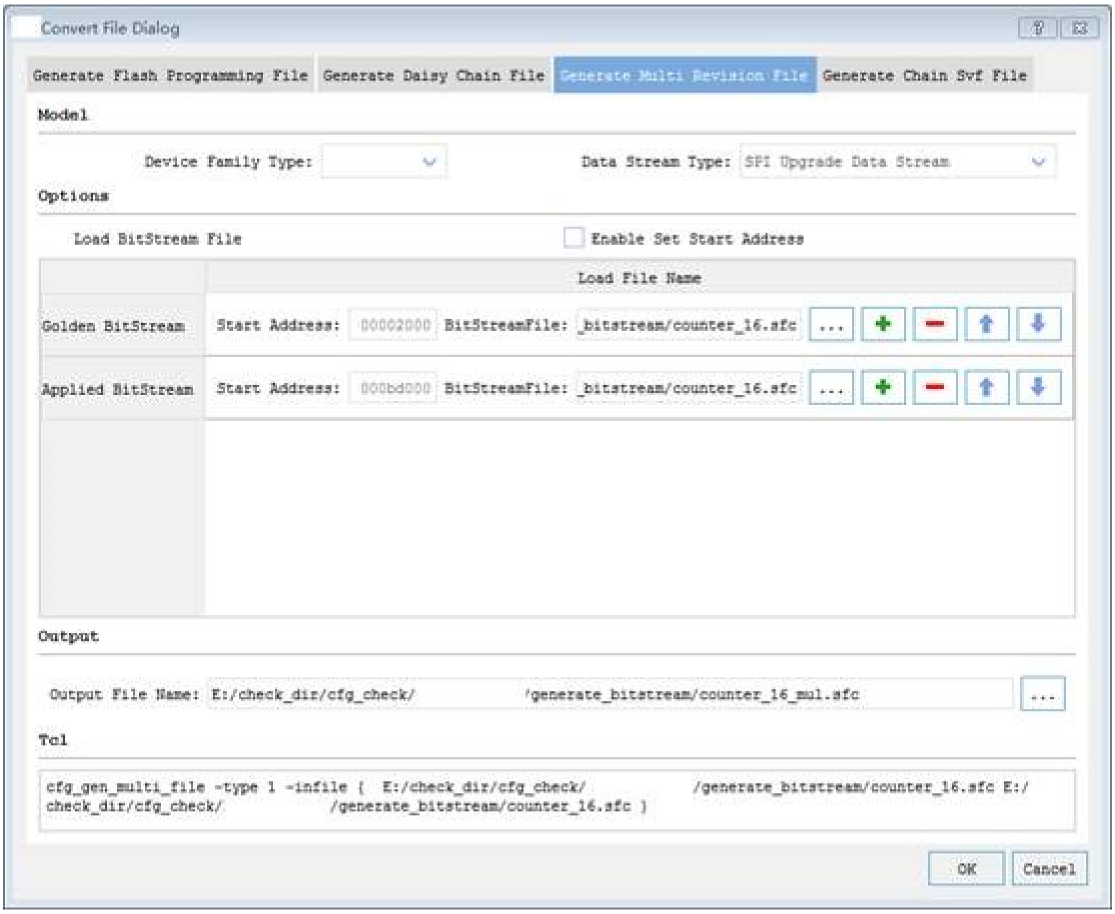


Figure 4-41 Generate Multi Revision Programming File

3.1 Generate operation process

Figure 4-41, click the button to add the SFC file that generates the golden combination data stream, the serial number in front of the file is the actual revision position, adjust and download to the corresponding revision position in Flash by moving the button. Output File Name is the name of the generated data stream. The default is to add _mul.sfc after the last added bit stream file name. You can also use the select file button to specify the file name.

After selecting the bit stream file and adjusting the location of the file, if the user does not need to set the address of the bit stream, Then click the OK button to generate the Multi Revision data stream.

3.2 Starting address of Logos series and Logos2 series configuration data flow

When generating data streams for devices of the Logos, Logos2 and Titan2 series, the software supports modifying the starting position of each added data stream. If the user needs to modify the start address of the data stream, then the start address of all data streams should be calculated and filled in, and the data stream cannot be modified only, otherwise the data stream cannot be generated, as shown in Figure 4 -42.

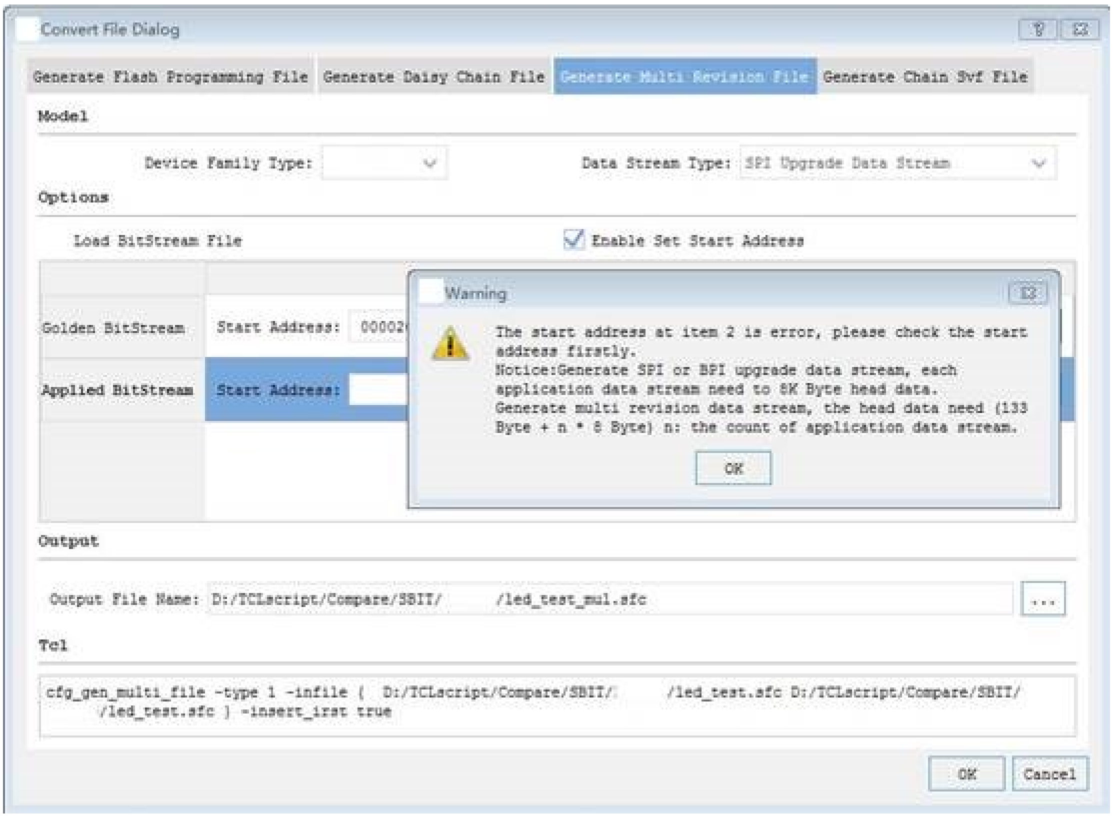


Figure 4-42 Address input is empty and cannot be generated

After checking Enable Set Start Address, you can enter the start address value of the data stream in the Start Address input box. Since a bit stream must be stored from the beginning of the sector, the value of the input start address must be an integer multiple of 4K (the sector size is 4K Byte). If the input value is not an integer multiple of 4K, after the input is completed, the software will automatically prompt whether the input value does not meet the specification and whether it needs to be automatically corrected. If you click No, the input value will be marked in red and modified to a value that meets the specification. The data flow can be generated later, as shown in Fig. 4 -43.

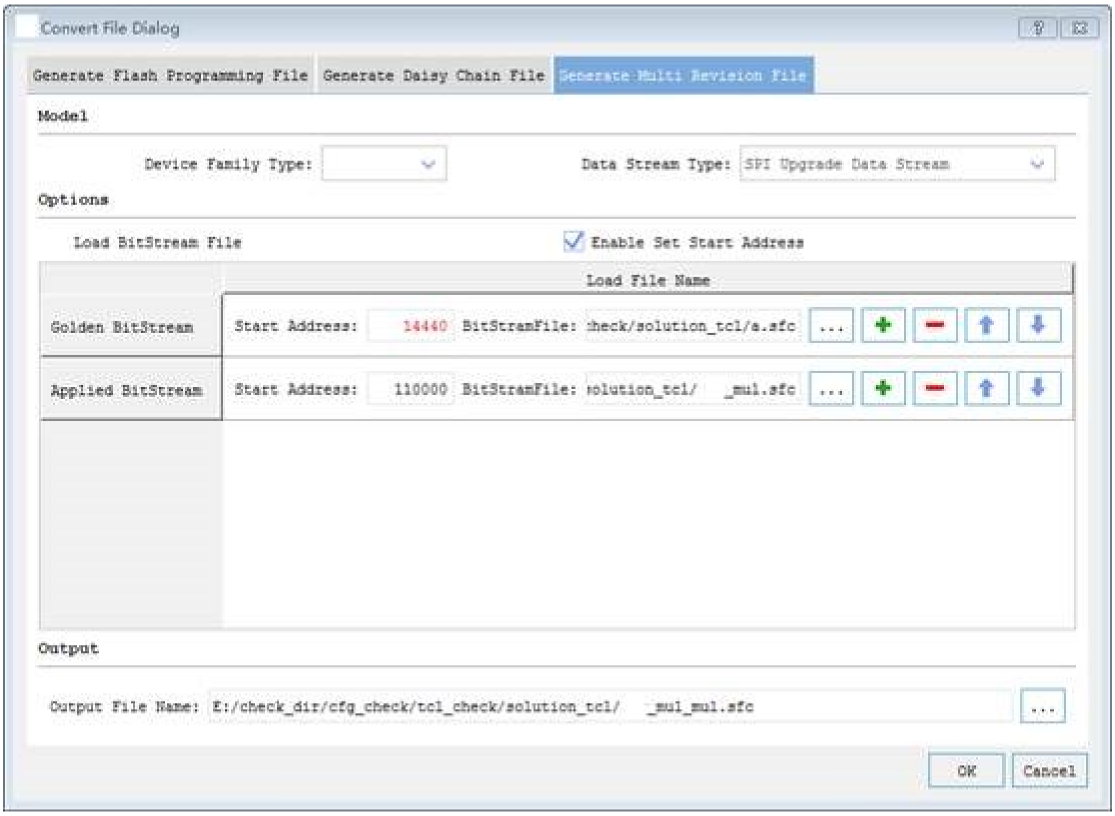


Figure 4-43 address input is not standardized and cannot be generated

When calculating the address of the data stream, the header data that should be added to generate the data stream should be considered. As shown in Figure -444, each type of combined data stream is composed of header data and user's bit stream data. When generating, the software will automatically add 108 bytes of desynchronization to each bit stream data The data constitutes a new bit stream data, that is, each bit stream is 108 bytes more than the original bit stream, and this part of the address should not be ignored when calculating the starting address.

Note: The header data referred to here refers to the header data of the entire generated data stream, which should be distinguished from the bit stream header of each data stream.

When generating the SPI Update Data Stream data stream and the BPI Update Data Stream data stream, each application data stream needs to add an additional 8KB of header data. For example: Now there is a golden data stream with a size of 800KB, and two application data streams with a size of 800KB. Then the size occupied by the header data is 16KB, so the starting address of the golden data stream is at least 0x4000 (16KB=0x4000). If the starting address of the golden data stream is 0x4000, then the first application stream starts at an address of at least

0x4000 + 0xC8000 + 0x6C = 0xCC06C(800K=0xC8000, 108 = 0x6C). Because a data stream must be stored from the beginning of the sector, it should be rounded up to 0xCD000, that is, the starting address of the first application data stream is at least 0xCD000.

Similarly, the start address of the first application data stream is 0xCD000, then the start address of the second application data stream is at least

0xCD000 + 0xC8000 + 0x6C = 0x19506C(800K=0xC8000, 108 = 0x6C). Because a data stream must be stored from the beginning of the sector, it should be rounded up to 0x196000, that is, the starting

address of the second application data stream is at least 0x196000.

Similarly, when generating a Multi Function Data Stream data stream, the header data needs to add an additional 133Byte + n*8Byte, n is the number of added data streams, and other start address calculation processes are the same as the SPI data stream start address calculation.

3.3 Logos series and Logos2 series data stream format

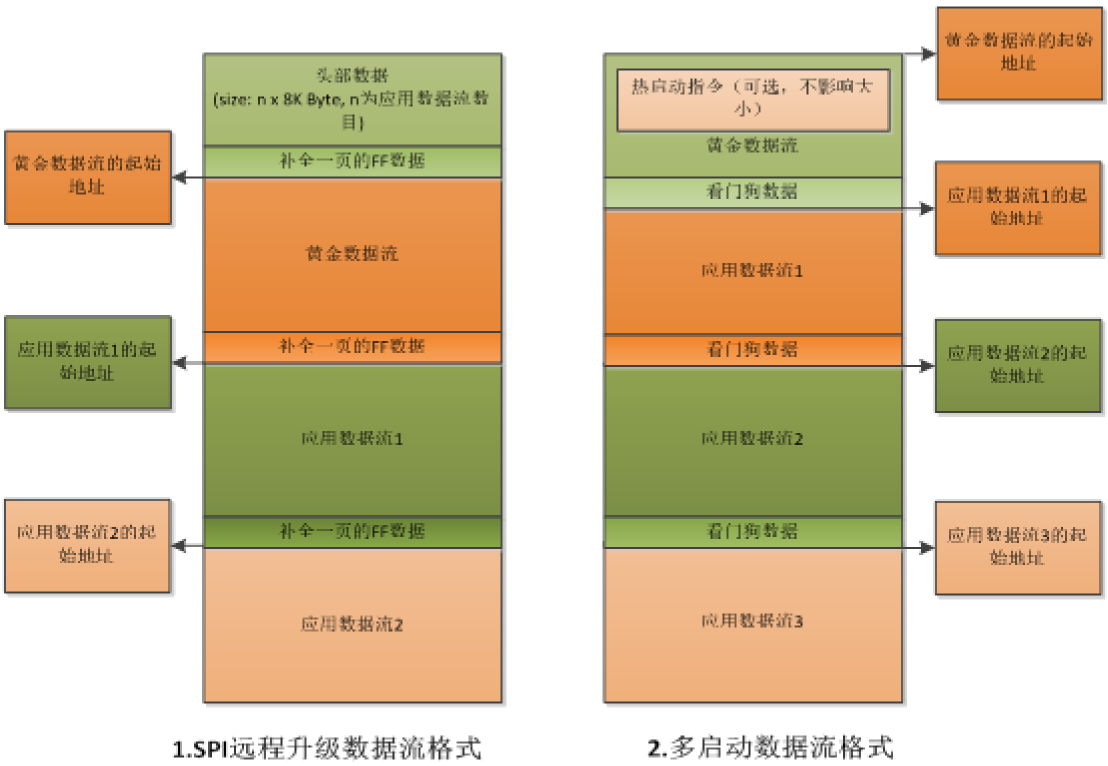


Figure 4 -44 Bit stream format for the combined data stream of the Logos, Logos2 and Titan2 families1

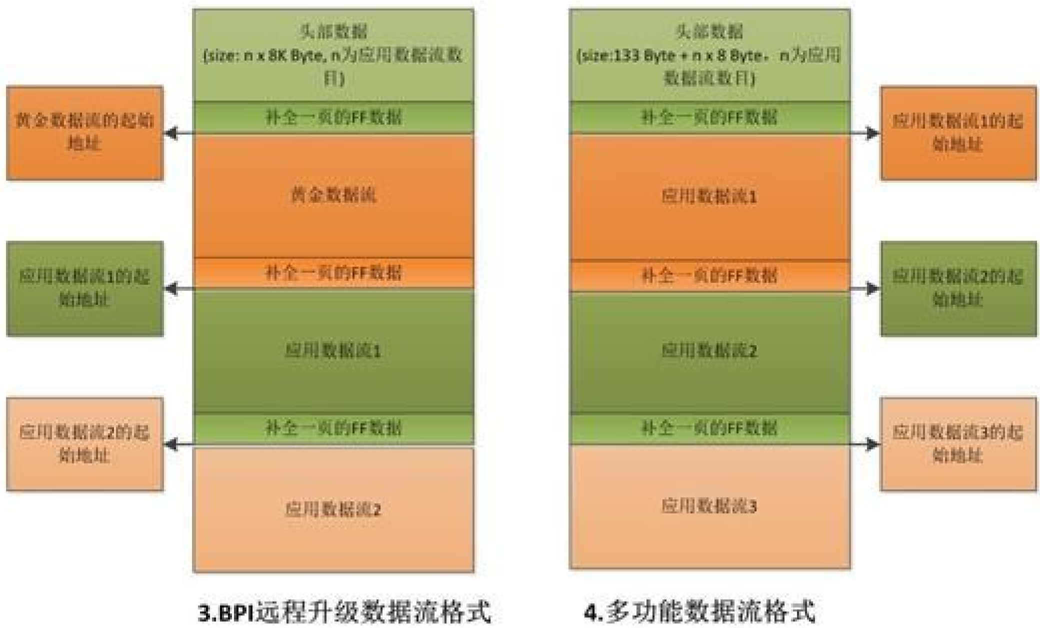


Figure 4-45 Bit stream format of combined data stream of Logos, Logos2 and Titan2 series 2

3.4. Start address of Compact series configuration data flow

Compact series devices generate the Remote Upgrade Data Stream and Dual Boot Data Stream data streams, the software supports modifying the starting position of each data stream, and the modification rules and constraints are roughly the same as those of the Logos series devices. But there are the following differences: ①. The Remote Upgrade Data Stream of the Compact series only needs to add 512Byte header data, and each data stream does not need to be synchronized before. ②. The golden data stream of the Compact series is formed by inserting 64Byte data on the basis of the common bit stream. ③. The Dual Boot Data Stream of the Compact series has no header data. Its data flow format is shown in Figure 4-46 below

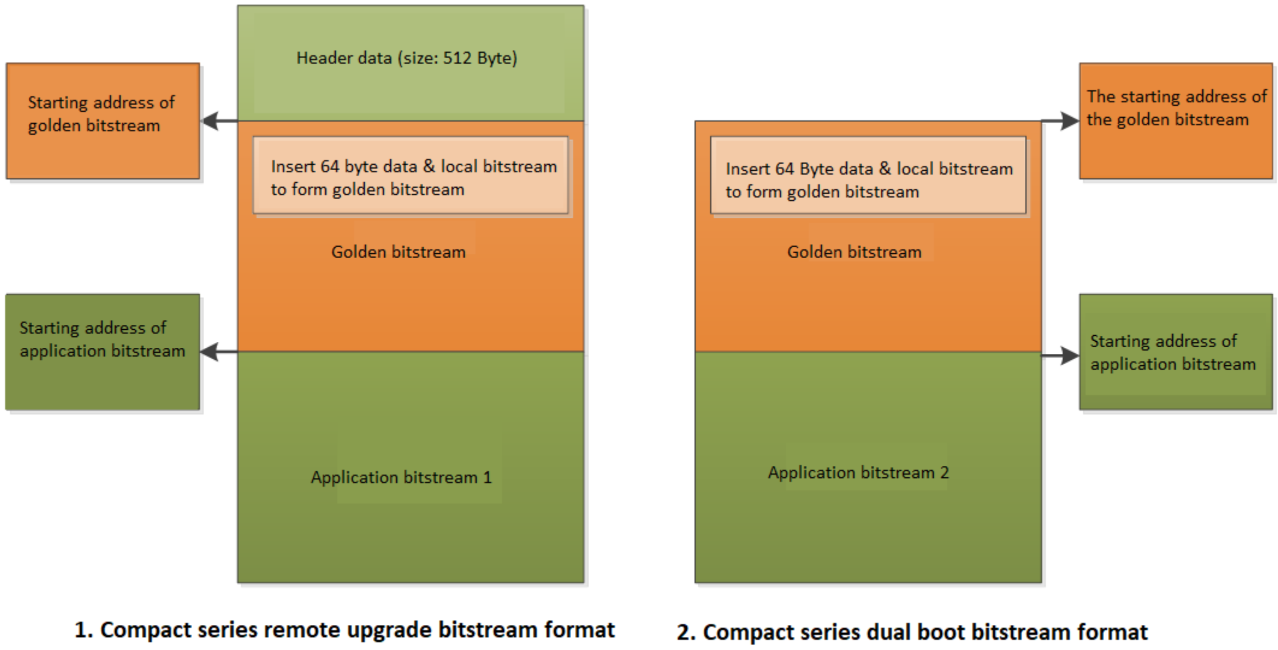


Figure 4-46 Bit stream format of Compact series combined data stream

4. Generate Multi/Dual Boot File (wizard mode)

This function generates solution-level bitstreams in wizard mode. After adding the device and connecting to the server, right-click the device, and in the pop-up menu, select the Generate Multi/Dual Boot File option. A window as shown in Figure 4 -47 will pop up (the content displayed in this window varies from device to device).

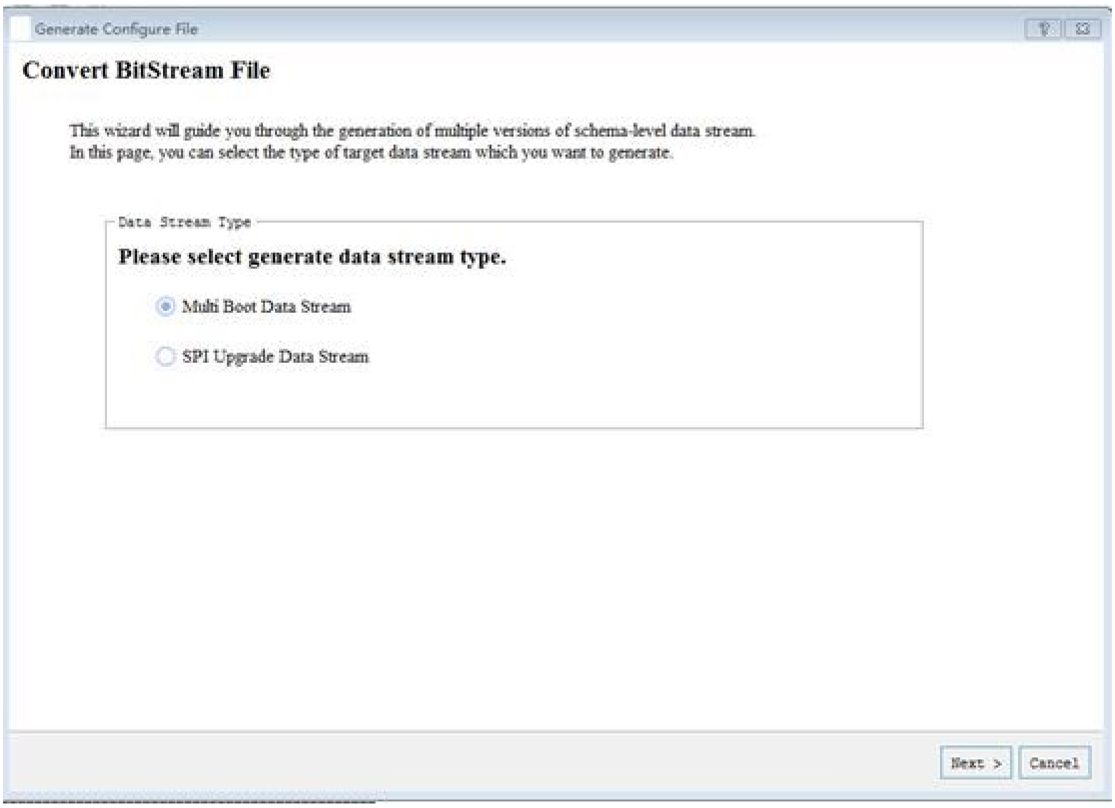


Figure 4 -47 PGL series program-level bit stream generation window

4.1 Master Auto and Dual Boot Data Stream

As shown in Figure 447, -this option is to generate a dual boot bit stream option.

Click Next to enter the dual-boot bitstream setting interface Figure 4 -48, set the golden data stream in Golden BitStream. Set up the application data stream in Applied BitStream. After setting, you can check the parts that need to be set at 51 in Figure 4, including setting feature control bits, generating -SVF files (refer to the corresponding chapters for related parameters) and separating and generating program-level bit streams.

Separately generate main and self-loading dual-boot bitstream function will split the main and self-loading dual-boot bitstream into _golden.sfc and _app.sfc. If generate svf is checked, _golden.svf and _app will be generated at the same time. svf (with address setting) two files.

Note: If you combine the functions of separate generation, svf generation, and feature control bit setting, the feature control bits will only be configured in the svf of the golden data stream in the separated svf. The feature control bits are not set in the svf of the application data stream.

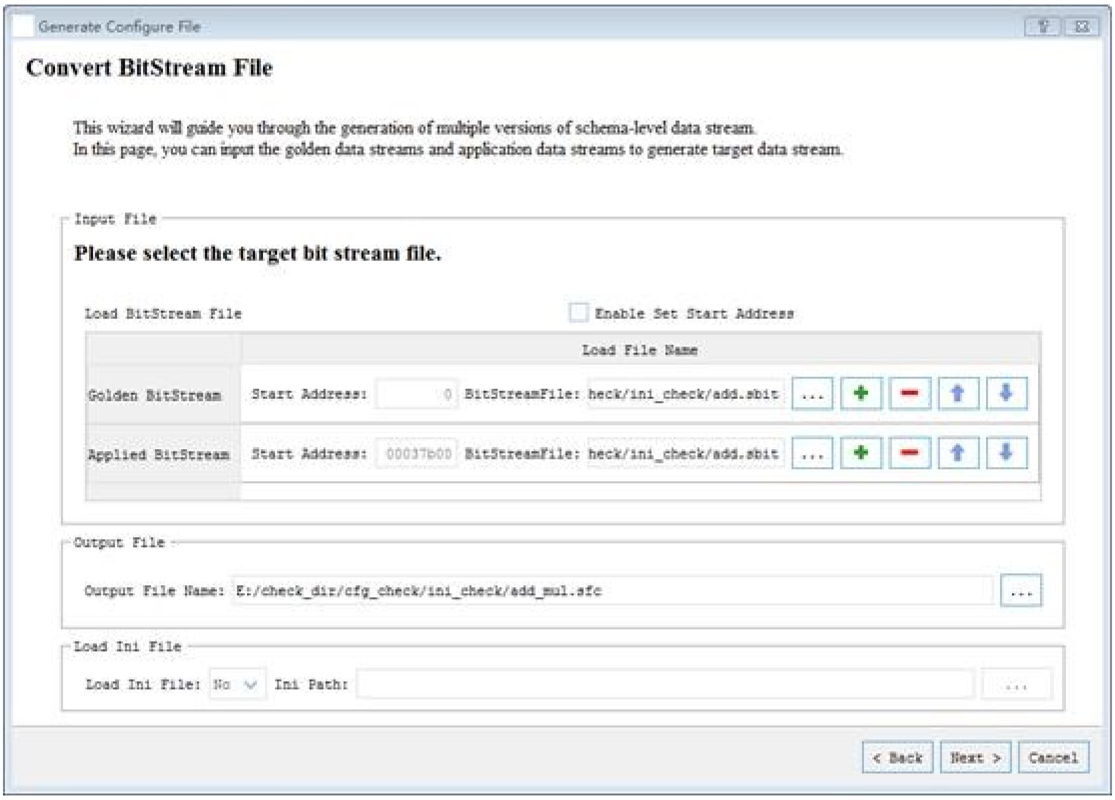


Figure 4 -48 dual boot bit stream setting interface

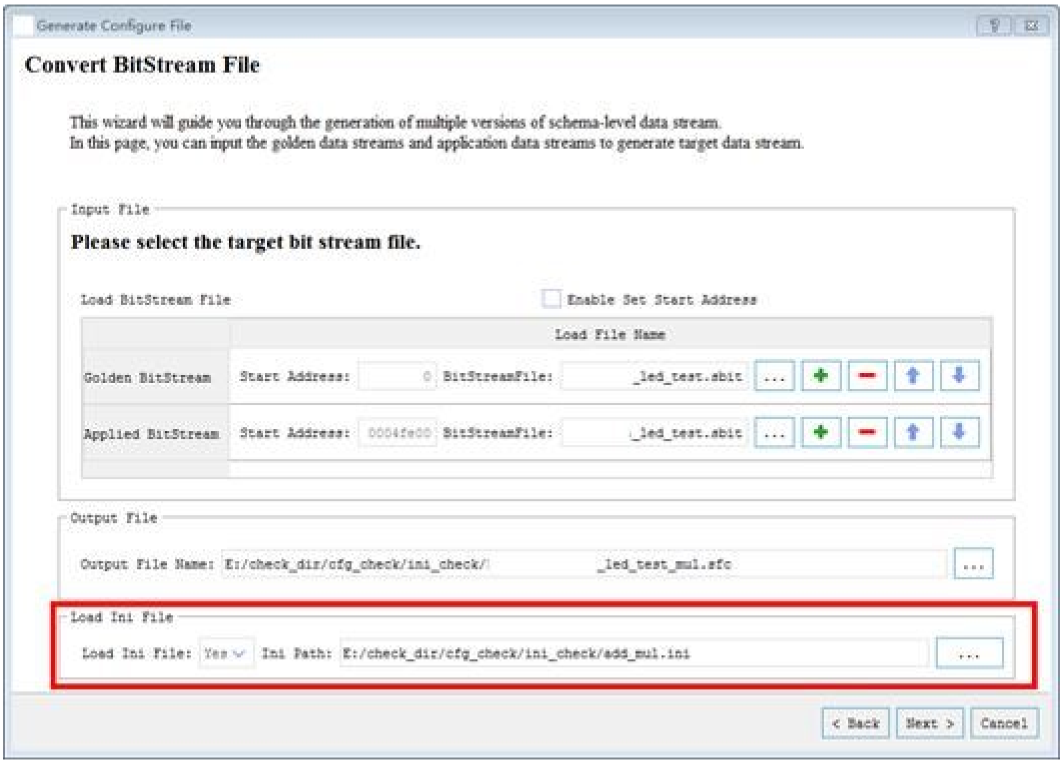


Figure 4 -49 configuration file import interface

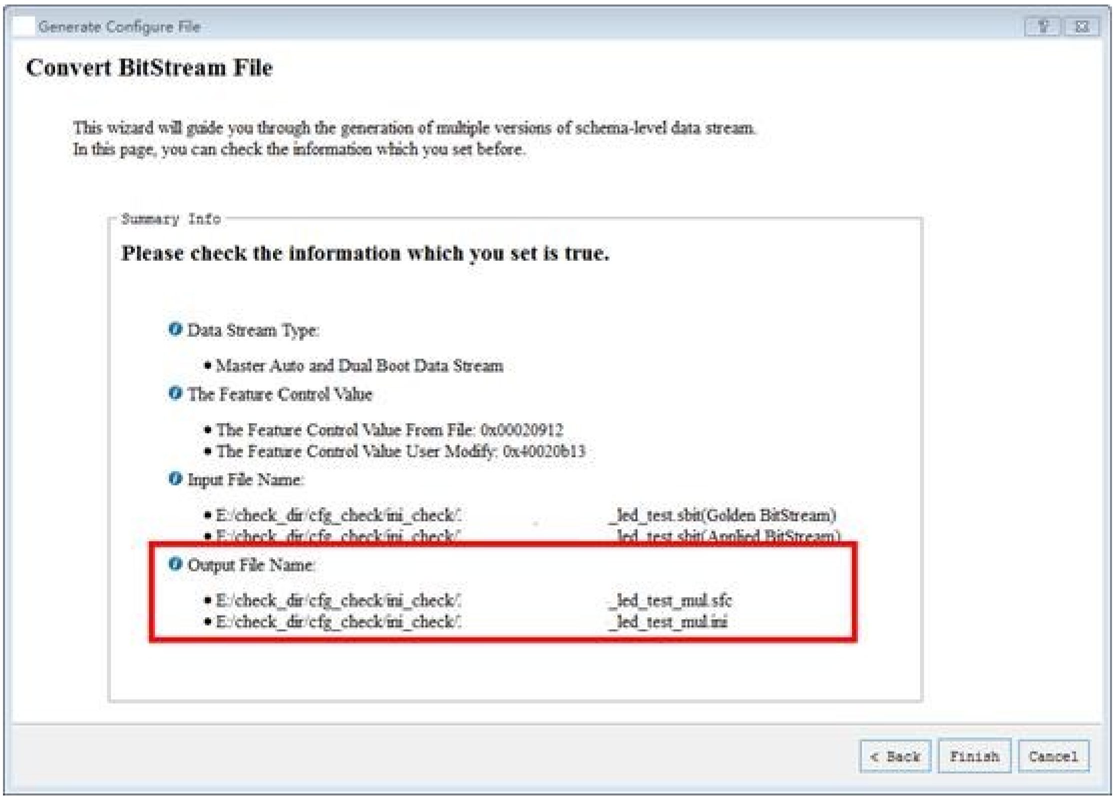


Figure 4 -50 configuration file generation interface

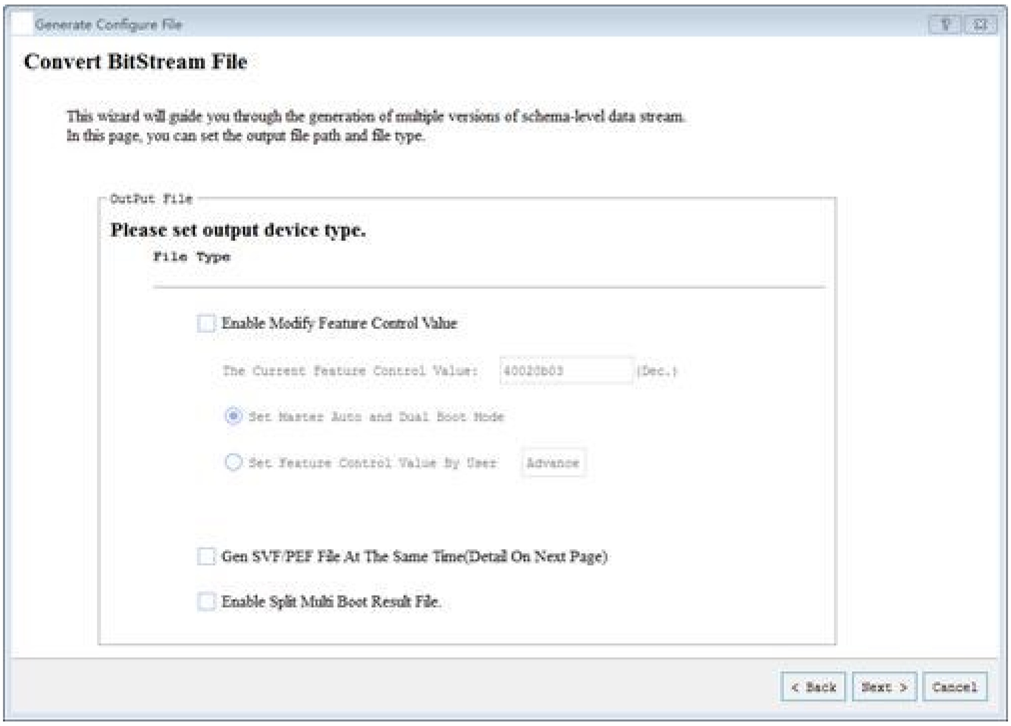


Figure 4 -51 dual boot bit stream and other settings

The scheme-level bit stream supports the use of configuration files (*.ini) to load settings, and users can import configuration files for subsequent parameter settings, as -shown in Figure 4 49. If the configuration file (*.ini) is imported, there is no need to set other parameters, and the wizard will directly jump to the summary page.

If the user does not have a configuration file (*.ini), the software will automatically generate a configuration file (*.ini) if the user chooses not to import the file (*.ini). The interface settings -are shown in Figure 4 48. The path of the automatically generated file (*.ini) will be displayed in the Output File column of the summary page, usually in the same directory as the generated Output File file (*.sfc), as shown in Figure 4-50.

next on the summary page, it will jump to the last page, which is the tcl prompt interface, as -shown in Figure 4-52. The prompt content can be copied into the tcl input box to generate a program-level bit stream for instructions.

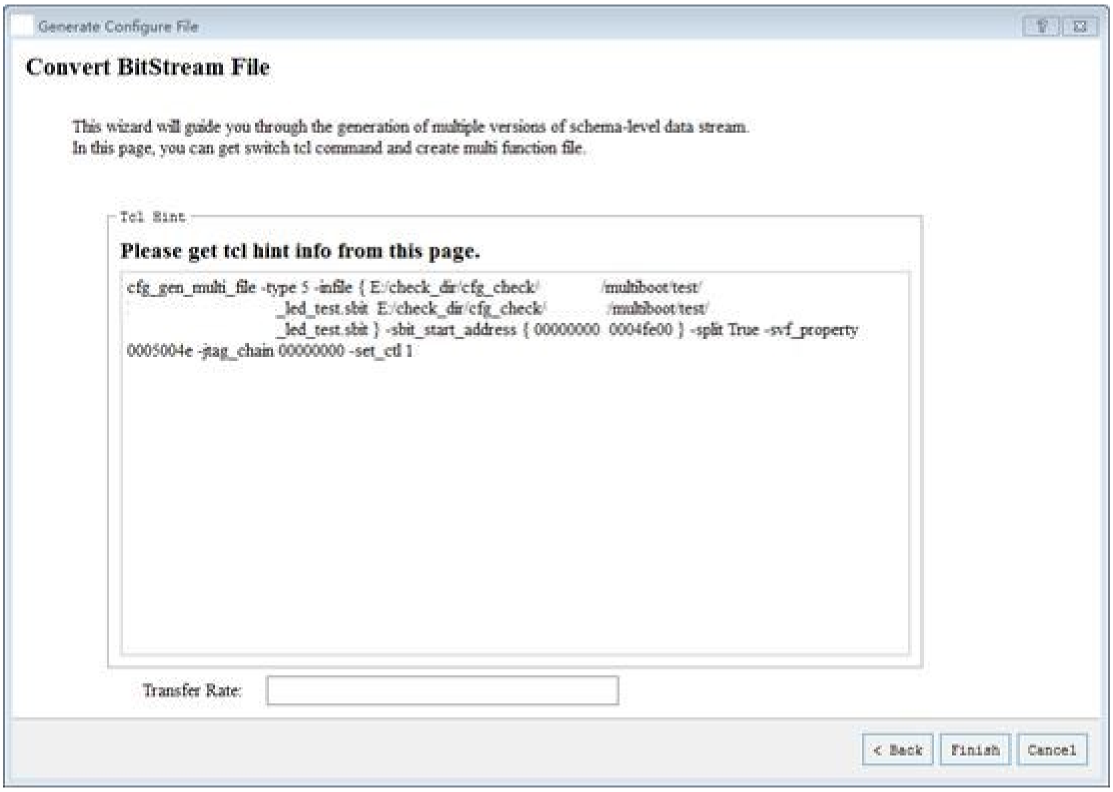


Figure 4 -52 tcl prompt interface

4.2 Multi Boot Data Stream

Multiple boot bit streams are available for PGL, PG2L series devices. Similar to dual-booting, the golden data volume and application bit stream are set first, and the IRST instruction can be inserted into the data stream (as shown in Figure 4 -53). Then set the output file path and output SPI mode (as shown in Figure 4 -54).

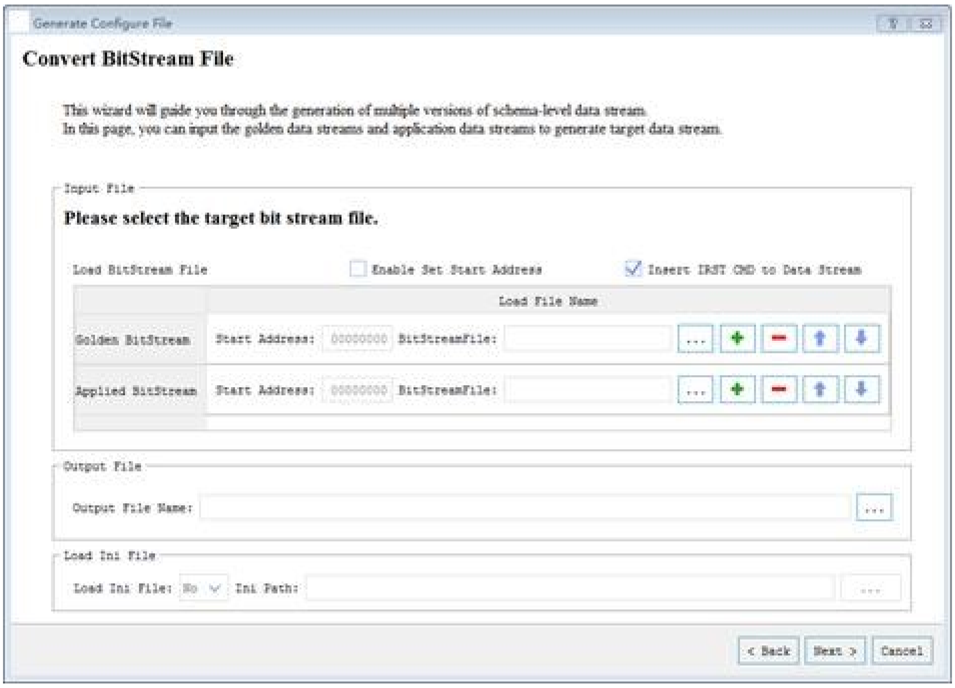


Figure 4-53 multi-boot bit stream input setting interface

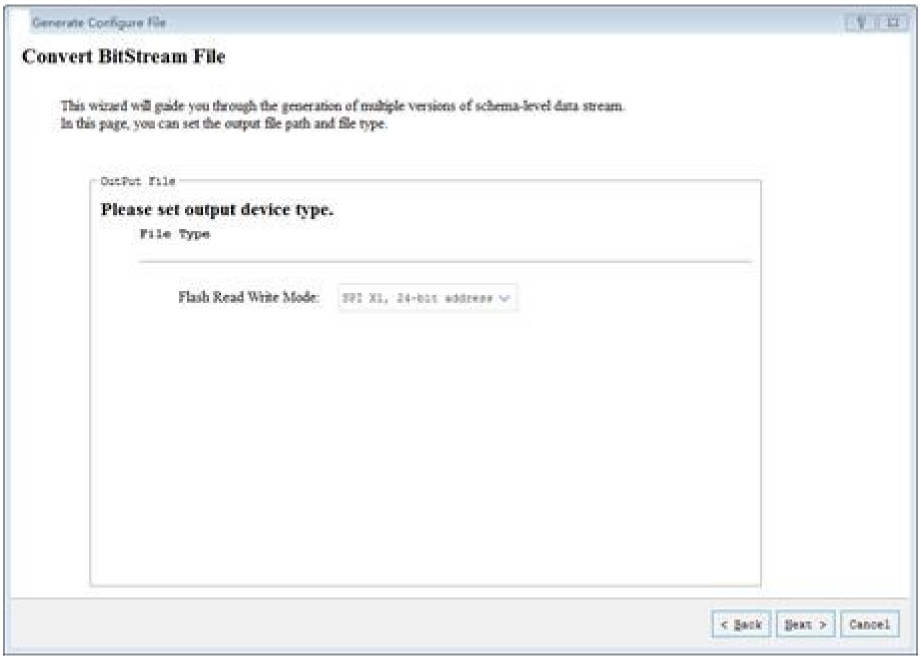


Figure 4 -54 multi-start bit stream output setting interface

4.3 SPI Upgrade Data Stream

The SPI upgrade data flow is consistent with the multi-boot bitstream setting interface. It is also enough to set the input golden data amount and apply the bitstream to set the output file path and output SPI mode.

5 Svf file merge

This function is mainly to generate cascaded svf/pef files. as shown in the Figure As shown in 4-55, in the Generate Chain Svf File window, add the svf file that generates the cascaded data flow,

and the serial number in front of the file is the location of the actual FPGA chain. The user can adjust the position through the buttons. It must be noted that the device corresponding to each data stream that generates the cascaded data stream must correspond to the device position downloaded to the actual FPGA chain.

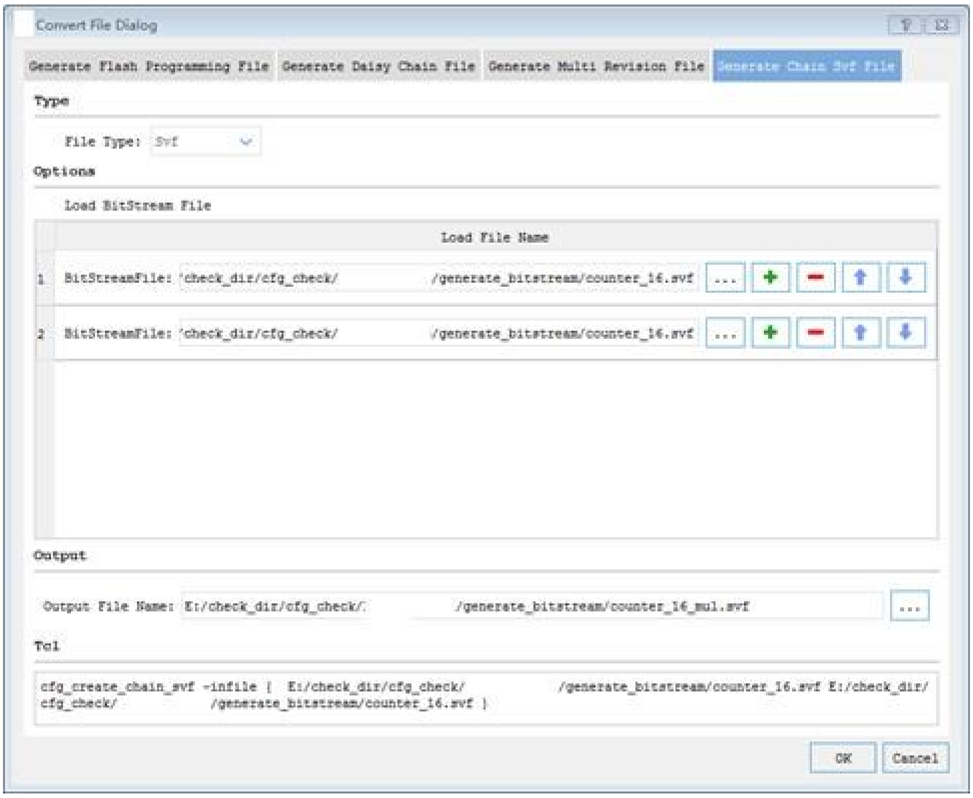


Figure 4-55 Svf file merge function

After selecting the svf file and adjusting the location of the file, click the OK button to generate the cascaded svf. If too many svf files are added, it will take a long time to generate and you need to wait for a while.

4.3.21 Jtag Common Configuration File (SVF & PEF)

SVF is Serial Vector Format, which is a part of IEEE1149.1. SVF has been used by chip manufacturers as the JTAG data exchange standard. The content of the SVF file is in ASCII text format, including operation commands and data, and the configuration of the programmable logic device is realized by controlling JTAG to send and receive data. Applicable to all Pango devices, it can be used to program, erase, verify embedded Flash or configure CRAM.

The PEF (Pango Embedded File) file is a compressed version of the SVF file, and it is a binary format file customized by Unigroup. It needs to use the driver provided by Unigroup. Applicable to all Pango devices, it can be used to program, erase, verify embedded Flash or configure CRAM.

1. Generate SVF file

Fabric Configuration supports the user to generate an executable SVF file with one click. Right-click on the selected device and select the One Step Create SVF option in the pop-up menu. After clicking Execute, it will appear as shown in Figure 4. -58 Set SVF File parameters and the path to generate the SVF file, click OK to generate the SVF, and the PEF file will be generated synchronously by default. If you do not need to generate a PEF file, you need to check Enable Default Not Create Pef

in Edit->perference in the menu bar, as shown in Figure 4-56.

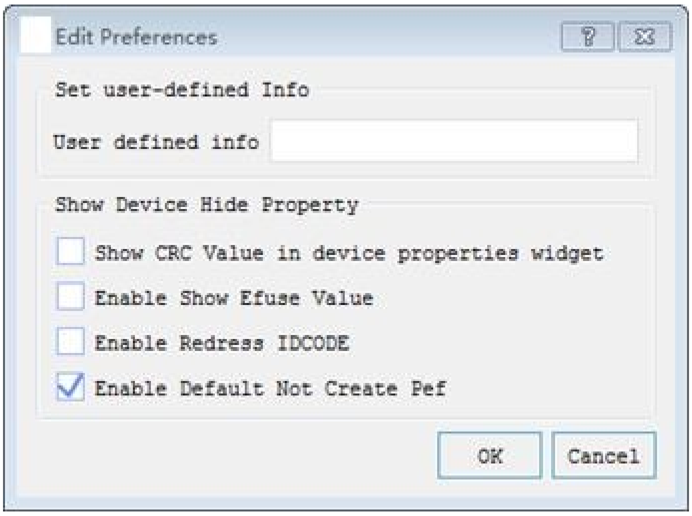


Figure 4-56 needs to generate pef

1.1. Generate Cram -type SVF files

Generate the svf file for cram, as -shown in Figure 4 57, you need to set the relevant parameters and the path of the SVF output file, and click ok. After the user uses it for the first time, he can copy the instructions in the tcl prompt box, so as to quickly generate svf later. The relevant parameters are introduced as shown in Table 4.

Table 4 CRAM SVF parameters

Enable Create Flatten SVF	Whether to generate SDR/SIR filled chain information SVF (note: use this function will not be able to use pef)
Enable Write Header and Comments	Whether to enable file headers and comments, ticking means enabling, default ticking
Disable Stopover Test Logic Reset	The bitstream configuration process prohibits jumping to the TLR
Enable SEC option of RUNTEST Command	The RUNTEST command is used to specify the waiting time that needs to be kept when the JTAG state machine jumps to a specific state (the specific waiting time is related to the operation option). There are two ways to realize the waiting time, TCK clock beat and system delay function : the When the option is checked, the waiting time is realized by the system delay function; when the option is not checked, the waiting time is realized by beating the TCK clock, which is checked by default
Set SVF Config Frequency	Indicates to set the configuration frequency of SVF, the default is 1MHz. When the "Enable SEC option of RUNTEST Command" is checked, the waiting time of the RUNTEST command has nothing to do with the frequency; when the "

	Enable SEC option of RUNTEST Command" is not checked, the number of TCK beat clocks corresponding to the RUNTEST command is proportional to the frequency, In this scenario, the TCK frequency on the user board needs to be the same as the frequency set by the software, or lower than the set frequency; if the TCK frequency on the board is higher than the frequency set by the software, the configuration may fail.

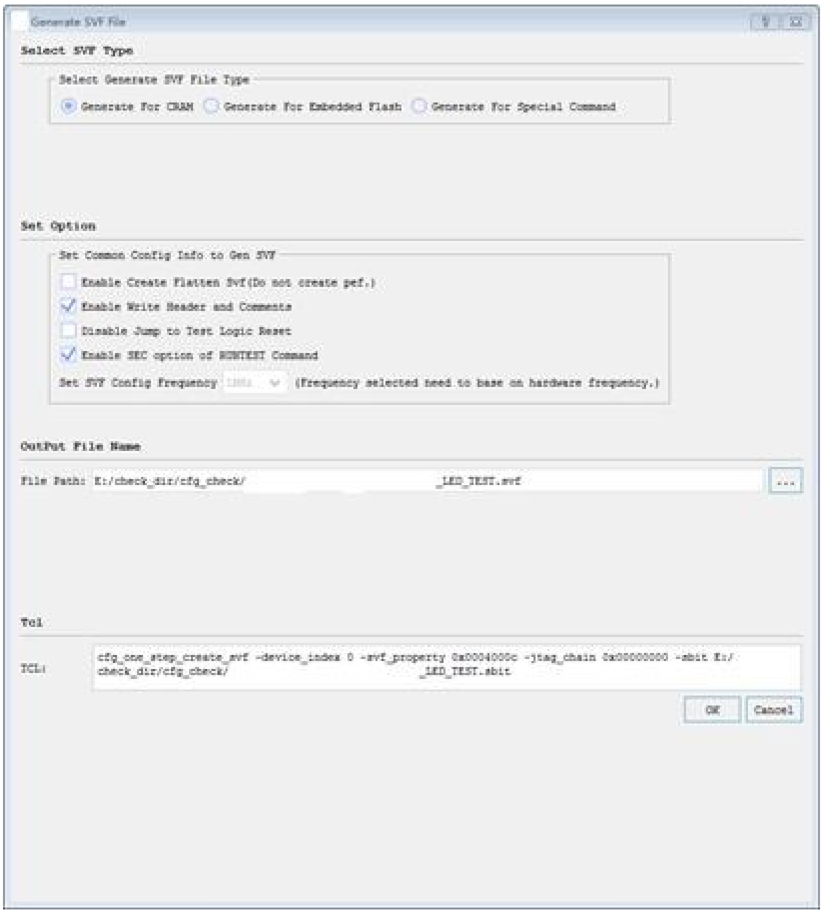


Figure 4-57 generates CRAM type svf

1.2. Generate SVF downloaded to embedded Flash

For the Compact series, users can generate SVF files configured with FPGA and SVF files configured with embedded Flash, which can be set in Embed Flash Properties in setting device properties. As shown in Figure 4.58 -, the Set Common Config Info to Gen SVF group box includes common SVF attribute settings, and Set Eflash Config Info to Gen SVF includes embedded Flash related parameter settings. Please refer to Table 5 for the meaning of specific parameters. After the user uses it for the first time, he can copy the instructions in the tcl prompt box, so as to quickly generate svf later.

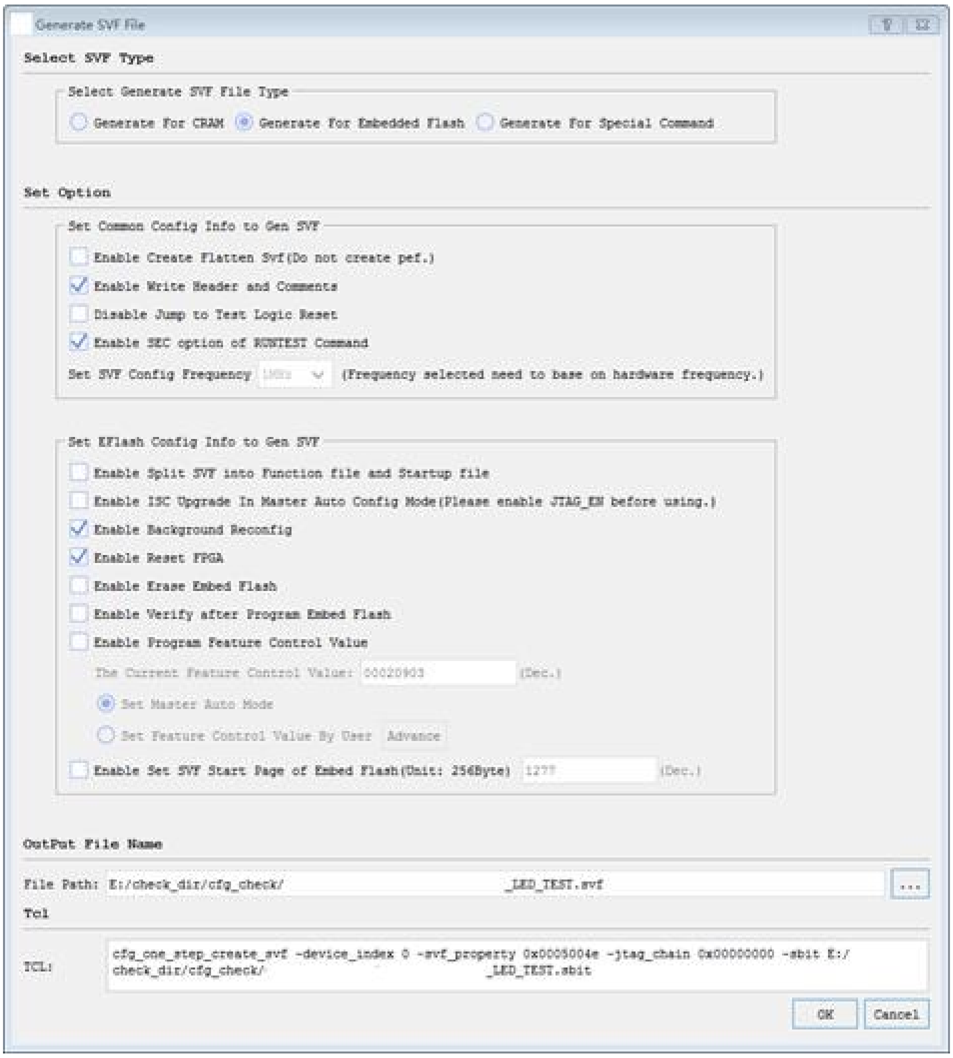


Figure 4-58 Select the generated SVF file path and parameters

Table 5 Embedded Flash SVF Configuration Parameter Meaning

Enable Create Flatten SVF	Whether to generate SDR/SIR filled chain information SVF (note: use this function will not be able to use pef)
Enable Write Header and Comments	Whether to enable file headers and comments, ticking means enabling, default ticking
Disable Jump to Test Logic Reset	The bitstream configuration process prohibits jumping to the TLR
Enable SEC option of RUNTEST Command	The RUNTEST command is used to specify the waiting time that needs to be kept when the JTAG state machine jumps to a specific state (the specific waiting time is related to the operation option). There are two ways to realize the waiting time, TCK clock beat and system delay function : the When the option is checked, the waiting time is realized by the system delay function; when the option is not checked, the waiting time is realized by beating the TCK clock, which is checked by default
Set SVF Config Frequency	Indicates to set the configuration frequency of SVF, the default is 1MHz. Enable SEC option of RUNTEST Command " is checked, the waiting time of the RUNTEST command has nothing to do with the frequency; when the " Enable SEC option of RUNTEST Command " is not checked, the number of TCK beat clocks corresponding to the RUNTEST command is proportional to the frequency, In this scenario, the TCK frequency on the user board needs to be the same as the frequency set by the software, or lower than the set

	frequency; if the TCK frequency on the board is higher than the frequency set by the software, the configuration may fail.
Enable Background Reconfig	Whether to enable background loading
Enable Split SVF into Function file and Startup file	To divide the SVF into function files and startup files, you need to configure the function files first, and then configure the startup files
Enable ISC Upgrade In Master Auto Config Mode	In-System Configuration) upgrade is enabled, and the IO port status will remain unchanged during the entire process of the remote chip upgrade. To use this function, it is necessary to ensure that the JTAG_EN pin led out of the chip is always pulled high externally during the remote chip upgrade process.
Enable Reset FPGA	Enable reset FPGA, enabled by default
Enable Set SVF Start Page of Embed Flash	Set the starting address of the SVF file
Enable Erase Embed Flash	programming the embedded Flash, which is disabled by default; this operation is not necessary, and the page erase operation is included in the process of programming the embedded Flash, enabling this operation will affect the SVF configuration time
Enable Verify After Program Embed Flash	Enable Verify after programming the embedded Flash
Enable Program Feature Control Value	Enable programming feature control bit
Set Master Auto and Dual Boot Mode	Only set the main self-loading dual-boot mode
Set Feature Control Value by User	Set the feature control bit, each bit can be set

1.3. Generate SVF with special instructions

As shown in Figure 4 -59, if you want to generate an SVF with special commands, you need to select the Generate For Special Command option in Select Generate SVF File Type. Only the Compact series supports generating SVF with special commands. After the user uses it for the first time, he can copy the instructions in the tcl prompt box, so as to quickly generate svf later.

The content of special commands included in Select Command to Generate is shown in Table 6.

Table 6 Meaning of special instructions

Reset FPGA Command	Reset FPGA
Check Done Bit of Status Register Command	Check whether the Done state is pulled high
Lock Embed Flash Command	Lock Embedded Flash
Entire Embed Flash Command	Erase the entire embedded Flash
Enable Verify after Erase Embed Flash	verify after erasing the embed flash
Enable Verify after Program Embed Flash	verify after programming the embed flash

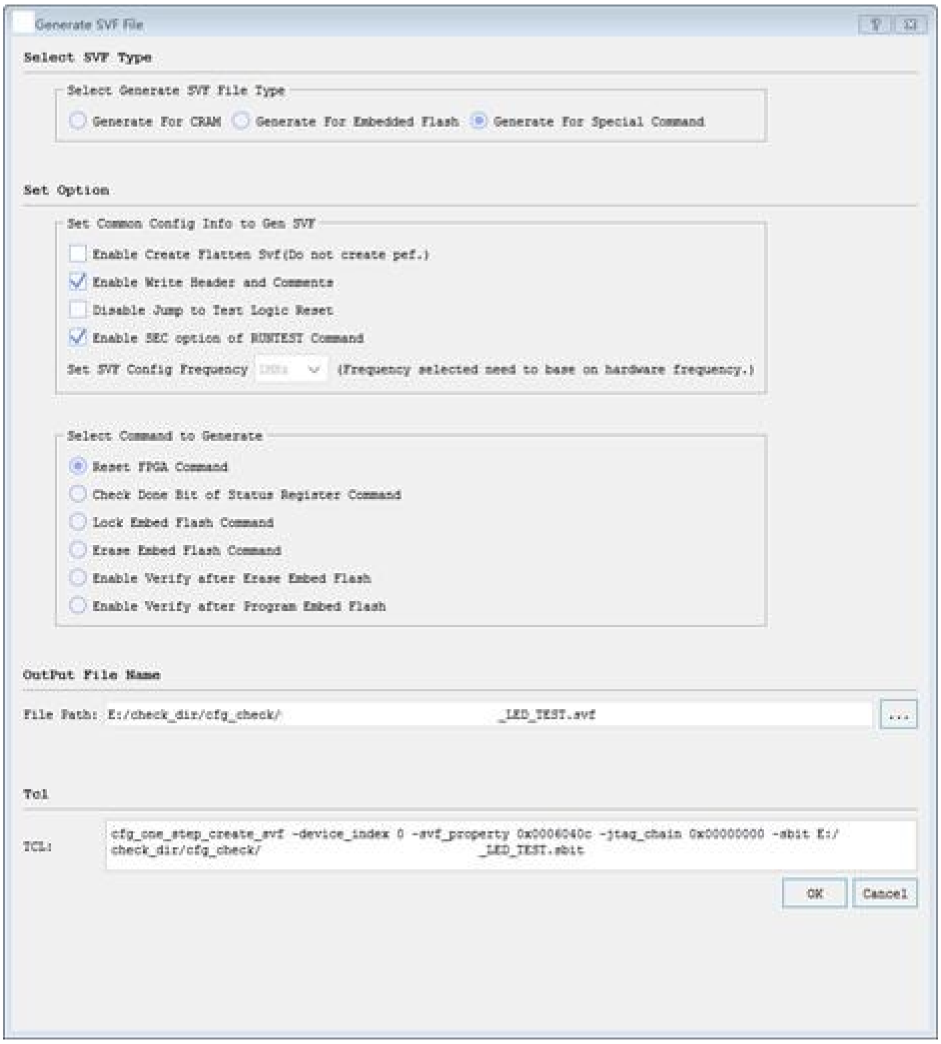


Figure 4-59 SVF with special instructions

2. Generate PEF file

PEF is a binary file compressed in a certain format for the SVF file. When selecting one-click to generate SVF, the PEF file will be generated synchronously, or use the Tcl command to generate it. The command is cfg_svf_to_pef. For details, refer to the Tcl command chapter.

3. Execute SVF/PEF file

First, connect to server (note: chain scan operation cannot be added), then right-click on the blank space of the display device, and select " -Add Pango Device " in the drop-down menu (as shown in Figure 4 10). In the pop-up selection box, select the SVF file or PEF file just generated, and after the addition is successful, as shown in Figure -460, at this point, the user can perform operations on the SVF. Right-click the device to pop up the menu, click the Execute SVF/PEF File option, and the software will execute the SVF file or PEF file.

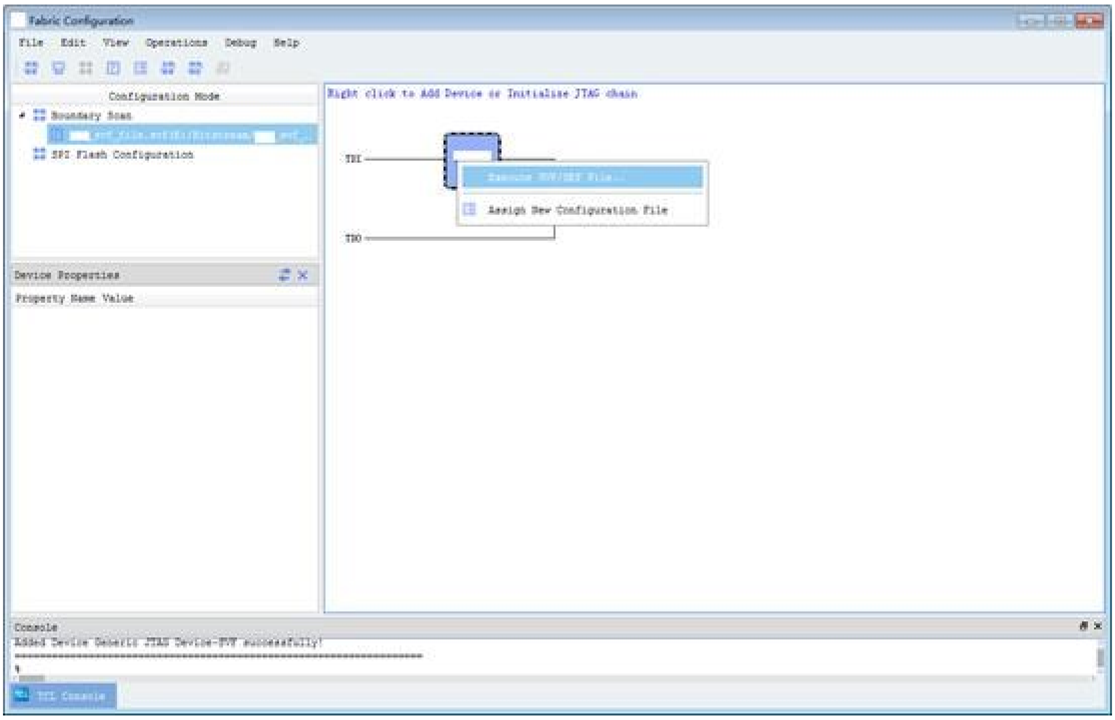


Figure 4-60 execute SVF file

4.3.22 Generation and Execution of Tcl Script

1. Record user operation function to generate Tcl script

In the Operations menu, as shown in Figure 461 -below, Tcl Record File records user operation functions in the form of tcl commands. Click the Create Tcl Record File menu option to create a new tcl file for recording, or click the Append Tcl Record File menu option to append to the In the existing tcl file, you can use the Execute tcl file function in the operation menu to execute the tcl file. When recording, the relative path is recorded in the tcl file, and the tcl_bit_file folder is created in the current path of the tcl file to save the bits operated by the user. For stream files, move the tcl file and tcl_bit_file folder to other paths at the same time, and the tcl file can also be executed.

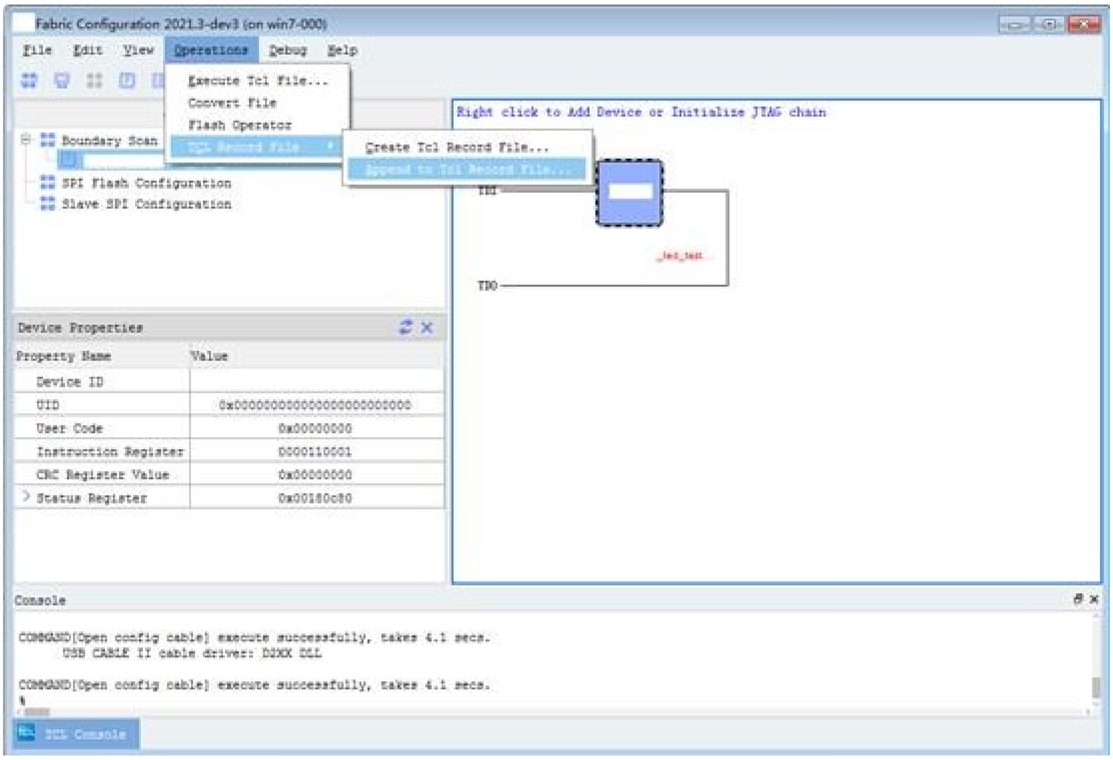


Figure 4-61 OperationsMenu

some user operations and Tcl commands are as follows

Table 7 Corresponding operations between user operations and Tcl commands

Tcl commands	user operation
cfg_connect	Connect to Jtag server
cfg_disconnect	disconnect Jtag server
cfg_add_device	Add FPGA device to boundary scan chain
cfg_assign_file	Configure Bitstream Files for FPGA Devices
cfg_program	Jtag programming FPGA device
cfg_readback	Jtag readback FPGA device
cfg_verify	Jtag verification FPGA device
cfg_read_device_property –mode 0	Jtag reads the ID of the FPGA device
cfg_read_device_property –mode 1	Jtag reads the UID of the FPGA device
cfg_read_device_property –mode 2	Jtag reads the user code of the FPGA device
cfg_read_device_property –mode 3	Jtag reads the status register of the FPGA device
cfg_remove_devices	the FPGA device on the boundary scan chain
cfg_flash_scan_device	SPI scan Flash device
cfg_flash_assign_file	Configuration file for SPI Flash device
cfg_flash_program	SPI programming Flash device
cfg_flash_erase	SPI Erase Flash Device
cfg_flash_readback	SPI readback Flash device
cfg_flash_verify	SPI verification Flash device
cfg_flash_remove_device	Delete SPI Flash device operation

Note the following points for this operation:

1> When the user operation is recorded as a tcl file, when the user operates the bit stream file such as operating the device configuration file, operating the SPI Flash device configuration file, etc., the user-operated file will be copied to the tcl_bit_file folder in the path where the tcl file is located

Next, the recorded path is also the path relative to the tcl file.

2> tcl records relative paths, so you can move the tcl files and tcl_bit_file folders to other paths or other computers for execution.

2. Execute the **Tcl** script

Fabric Configuration supports users to perform batch execution operations (Tcl commands). Click the Execute Tcl File option -under the Operations menu in the menu bar, and a window for selecting tcl scripts will appear as shown in Figure 4.62. Select the corresponding Tcl scripts to execute.

Fabric Configuration also supports starting and executing scripts on the shell side. When starting on the shell side, the Fabric Configuration interface will not be displayed. The specific operation method is: switch to the directory where the executable program is located, then start the shell terminal of the operating system, and enter the command "cdt_cfg.exe –ip target host IP –port target host port –file Tcl script absolute path ” For example: cdt_cfg.exe –ip 127.0.0.1 –port 65420 -file E:\pan\project\sbit\ PGL50H _tcl_script.tcl.

Note: For a tutorial on using Tcl commands, see the introduction to Tcl commands in the fourth section of this chapter.

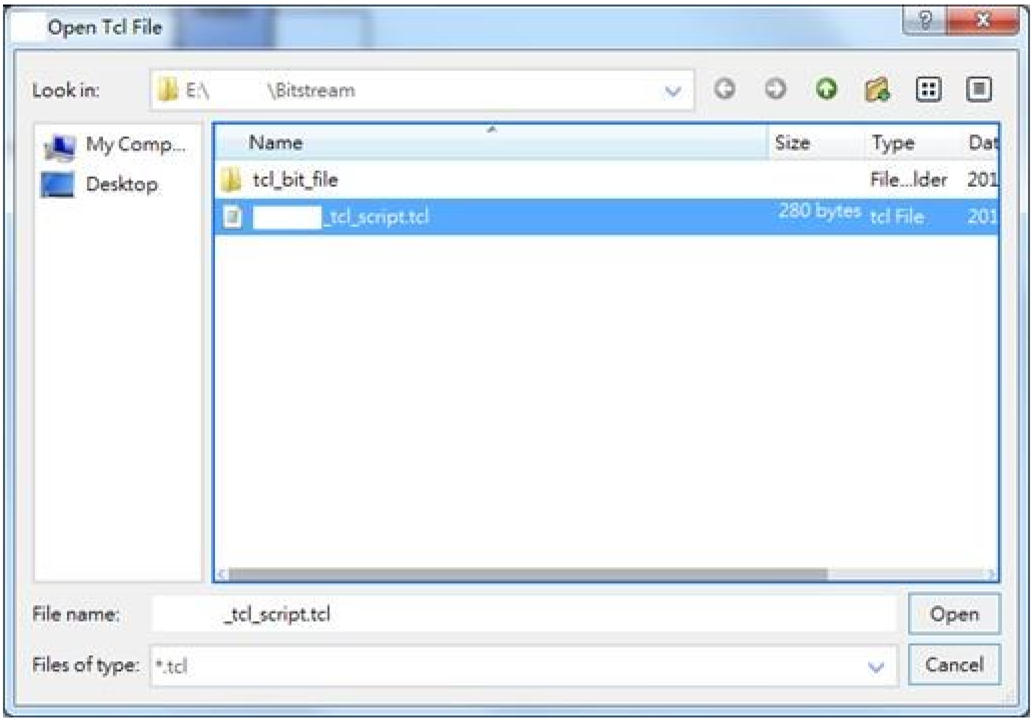


Figure 4-62 execute tcl script window

4.3.23 Set user-defined information

Click Preference under the Edit menu, In the dialog box that pops up, set the user-defined information, as shown in Figure -463. The user controls whether to display the CRC value of the bit stream in the device properties window and set the user-defined software title.

When multiple configuration software is opened, the user can set identification distinguishing information to distinguish different uses of the multiple configuration software.

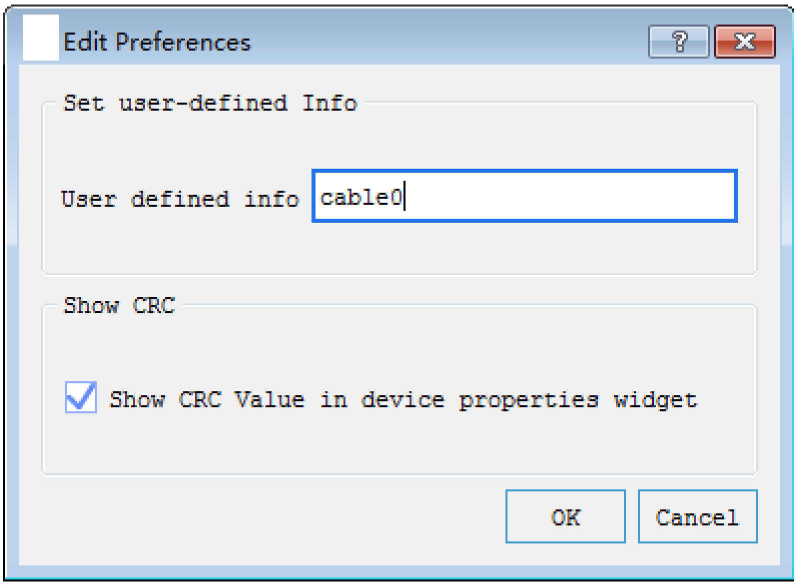


Figure 4-63 Preference dialog box

After the user-defined information is successfully set, the user-defined information will be displayed in the title bar, as shown in Figure -464. When the user executes the corresponding tcl script, a tcl script running log with user-defined information as the file name will be generated, as shown in Figure 4-65 below.

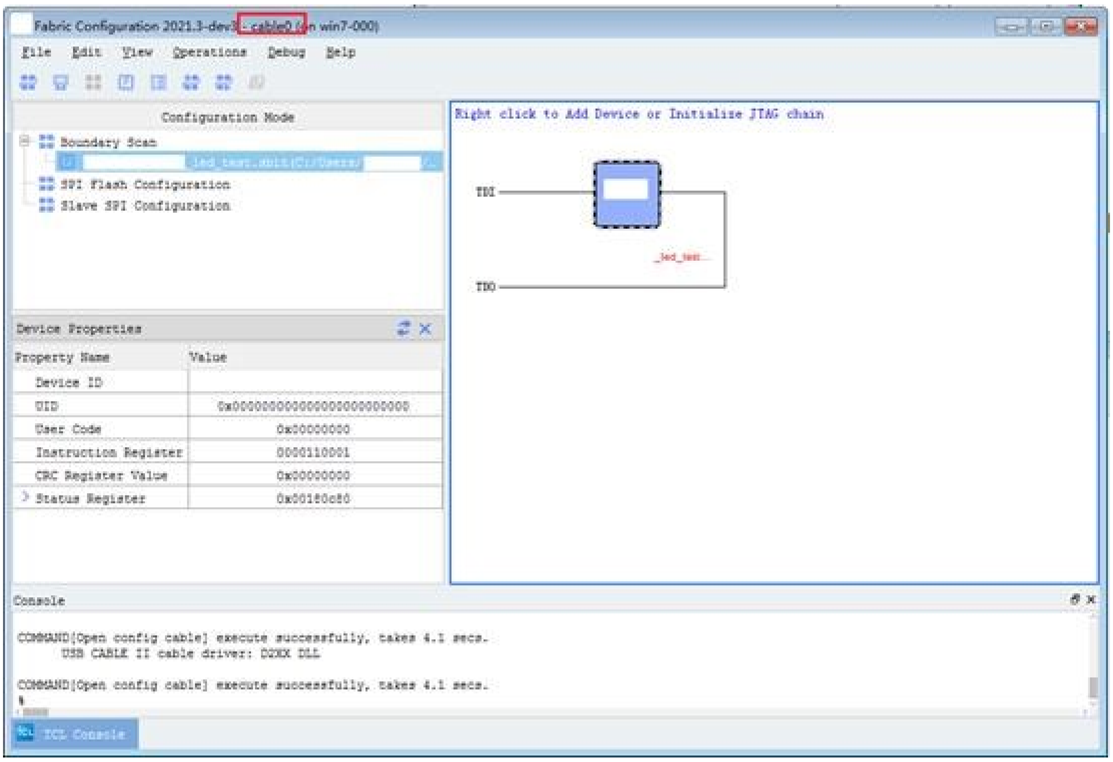


Figure 4-64 user information in the title bar



Figure 4.65 -log file after executing Tcl

4.3.24 Chain Integrity Test

In the Debug menu, click the Chain Integrity Test option to test whether the currently connected Jtag chain is in the state of the path. If the currently selected Jtag chain is in the state of access, it will display the number of devices on the current chain, as shown in Figure 4-66.

```
begin to test chain integrity..
The total device number in the chain is : 1
Jtag chain integrity test succeed, takes 0.5 secs.
```

Figure 4 -66 Test chain integrity successful

If the test chain connectivity fails, it will be displayed as shown in Figure 4 -67.

```
begin to test chain integrity..
E: Configuration 0006: Fail to detect jtag chain device
E: Configuration 0014: Jtag chain integrity test fail, takes 0.0 secs.
```

Figure 4 -67 Failed to test chain integrity

4.3.25 Set user-defined Flash

Click the Flash Operator option in the Operations menu, and perform user-defined FLASH settings in the pop-up dialog box, as shown in Figure 4-68. The user selects the Flash Type to display all the currently supported devices or all the devices added by user-defined (only the devices added by user-defined can be deleted).

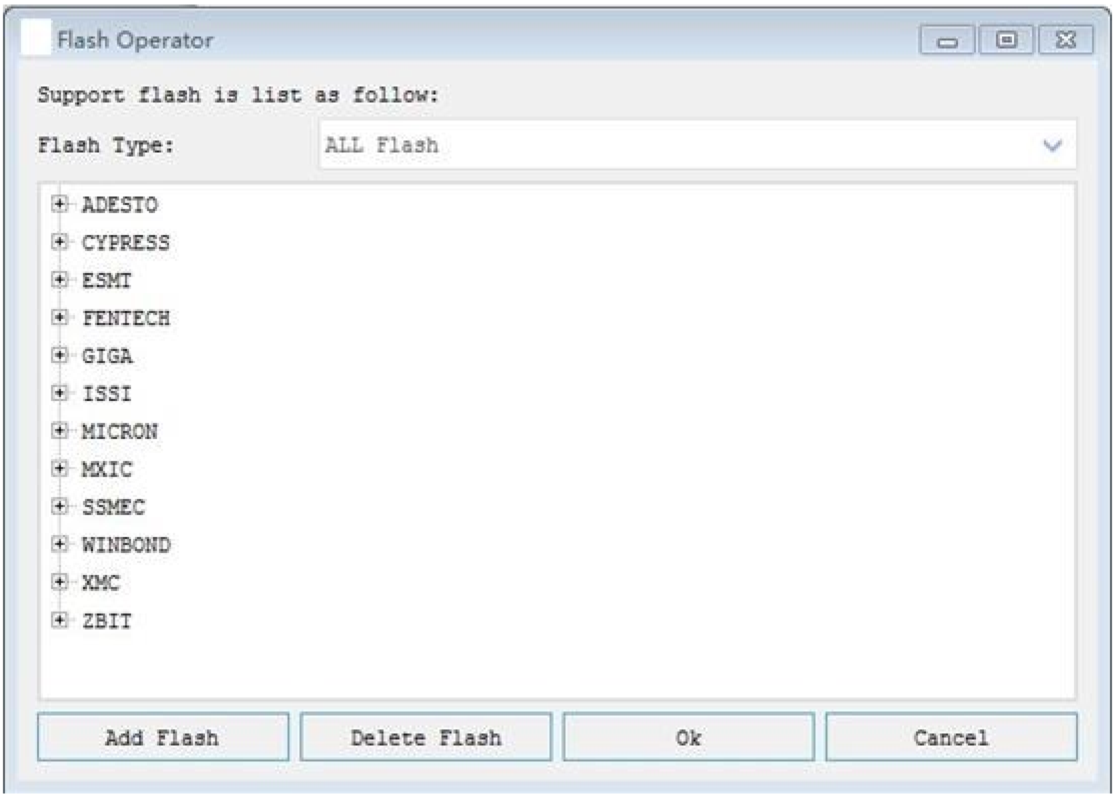


Figure 4-68 Flash Operator main interface

add new devices by clicking the Add Flash button. The adding interface is shown in Figure 4-69. First, select the manufacturer and series to be added, and click Next to enter the device specific information setting interface, including the device name and device ID (as shown in Figure 4-70). Finally enter the summary interface and click Finish to complete the addition.

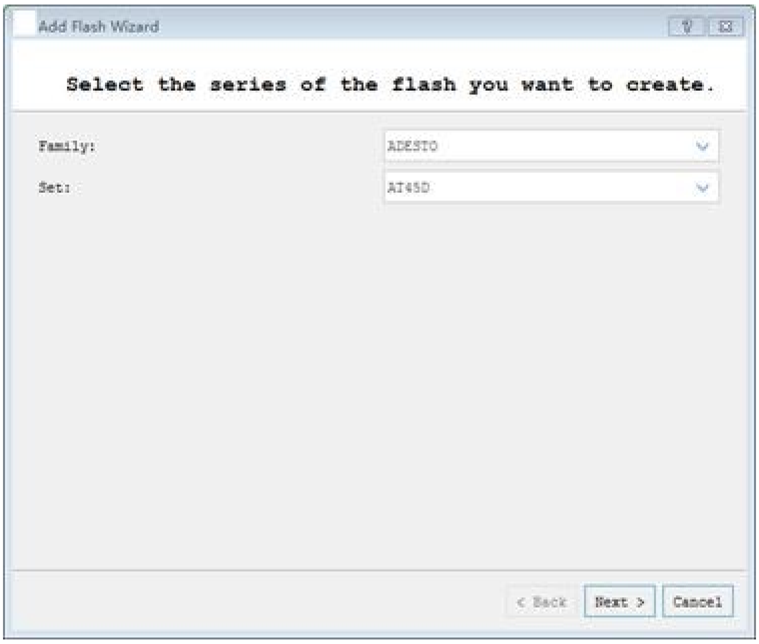


Figure 4-69 Added Flash manufacturer series selection interface

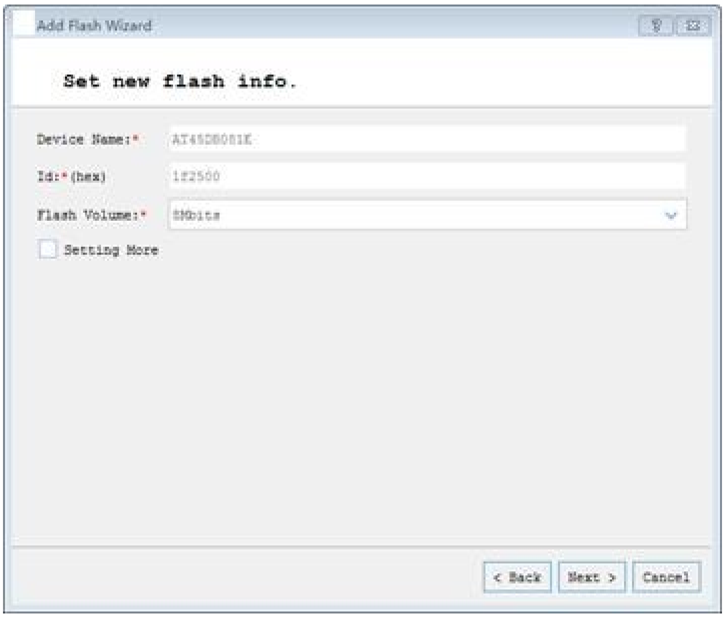


Figure 4-70 device information setting interface

After adding the Flash, click the Ok button to complete the real addition. Click Cancel or close the window to roll back the relevant addition.

4.4 Getting started with Tcl commands

4.4.1 Introduction to Tcl commands

The fabric configuration tool supports some native Tcl language commands such as cd, source, etc., and supports some custom Tcl commands of our company such as cfg_program, etc.

For a series of repetitive interface operations, users can input multiple Tcl commands or execute (*.tcl) files at one time on the Tcl console, and let the software automatically execute these operations to improve user work efficiency.

4.4.2 View all supported Tcl commands

"cfg_help" command on the console of the Fabric Configuration tool to view all Tcl commands -supported by the current version, as shown in Figure 4 71.

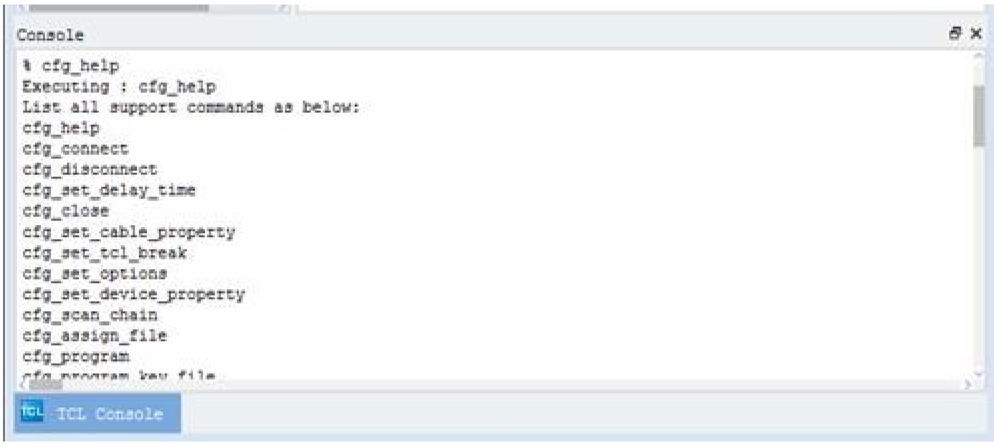


Figure 4 -71 cfg_help displays all supported Tcl commands

4.4.3 Query the parameters of specific Tcl commands

1. Console query

If the user wants to query the usage of a specific command, he can execute " cfg_xxx -help" on the console to print out the specific usage method of the command.

Enter " cfg_program_user_efuse -help " in the console as shown in Figure 4 -72 below, and you can find out that the command has three parameter options, among which the options with square brackets [] are optional, and those without square brackets [] are required. Each Tcl command has a [-help] option to facilitate users to query the specific usage of this Tcl command.

```
Console
% cfg_program_user_efuse -help
Executing : cfg_program_user_efuse -help
H: cfg_program_user_efuse : program 32 bits user efuse (Version 2021.4-dev1 build 82143M)
    [ -help  ]
        Turn on to print this message
    [ -device_index integer_number ]
        Specify set device's index
    -value string
        Specify set user's eFuse value.   ---Binary Number.
          Such as, 00000000000000000000000000000101.

Executing : cfg_program_user_efuse -help successfully.
%
```

Figure 4 -72 The specific usage of console query cfg_program_user_efuse

2. Query user manual

Users can also check the usage of a specific command by viewing (5) Tcl command specific parameter query in Chapter 4 of Fabric Configuration_User_Guide (Configuration User Manual).

Note: If the prompt command format is illegal, it means that the parameter format entered by the user does not meet the format requirements of the parameter option. The user can enter " cfg_xxx -help " to view the format requirements of the input parameters of the command.

When inputting Tcl commands, users should pay attention to the commands, options and parameters separated by spaces;

- " before the parameter option is the " - " in the English input method. If the user enters the Chinese " - ", it will prompt that the input command is an invalid command;

Tcl command shown in Figure 4 -72, the user only needs to enter the cfg_program_user_efuse -value 1010

4.4.4 Obtain Tcl commands for interface operations

to let the software automatically execute these operations by inputting multiple Tcl commands or executing (*.tcl) files to improve user work efficiency, but users may not know which ones at the beginning Which Tcl commands correspond to the operation, and the correct execution order of the Tcl commands (Note: When executing multiple Tcl commands, the wrong execution order will cause the execution of the Tcl commands to fail).

Therefore, the Fabric Configuration tool provides two methods to facilitate users to quickly obtain a series of Tcl commands corresponding to a series of interface operations performed by themselves. These two acquisition methods are very practical when users use Tcl commands.

1. Obtain from the Tcl box below the operation interface

Under the operation interface of some specific functions (such as svf/pef file generation, file conversion function, programming efuse, etc.), a series of corresponding Tcl commands will be generated in the -Tcl box in real time according to the parameters set by the user on the interface, as shown in Figure 4 73.

The user can either directly realize this function by clicking the relevant button on the interface,

or realize this function by copying the Tcl command in the box to the console and pasting it. For highly repetitive operations, users can save the generated Tcl commands, so that they can directly use Tcl commands to replace a series of interface operations next time.

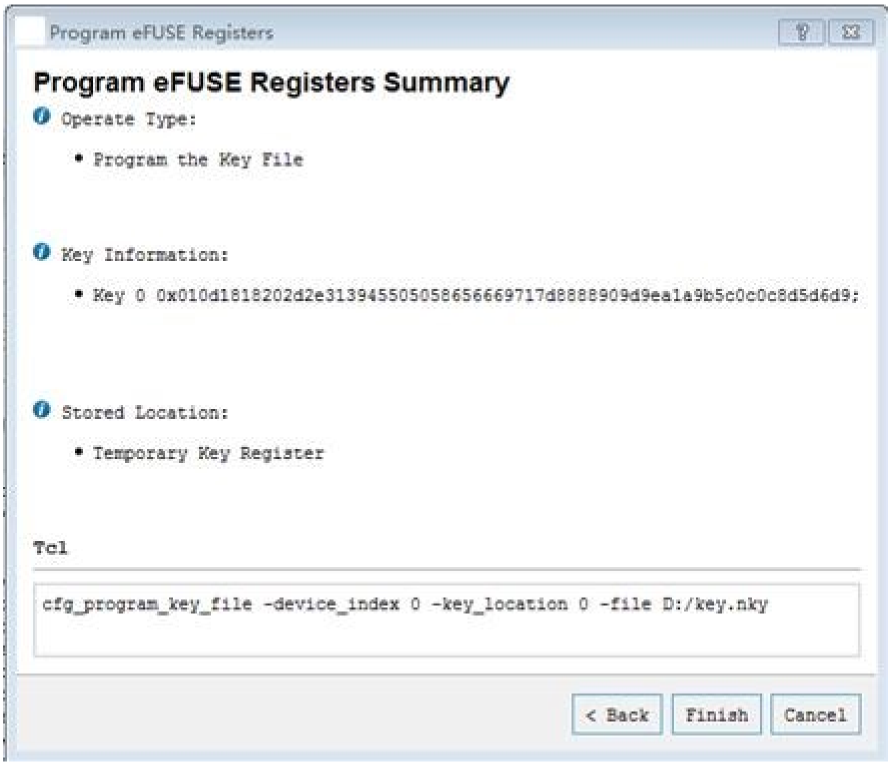


Figure 4 -73 Operation interface for burning temporary key files

2. Obtained from the tcl file generated by the Tcl record function

Operations - TCL Record File - Create Tcl Record File in the menu bar to create (*.tcl) files to record the Tcl commands corresponding to all subsequent interface operations of the user. (*.tcl) The Tcl commands recorded in the file are available for user reference. Please refer to 4.3.22 Generation and Execution of Tcl Script for details on the usage of Tcl recording function.

4.4.5 Executing Tcl commands

the user obtains the Tcl command, there are many ways to execute the Tcl command, and the user can choose the appropriate execution method according to the actual usage.

1. Execute under the console of Configuration software (including Windows version and Linux version)

There are four ways to execute Tcl commands on the console of the Configuration software :

Method 1: Enter or paste Tcl commands on the console, such as cfg_program_user_efuse -value 1010.

Method 2: Enter the source command and the address of the tcl file on the console, such as source D:/test.tcl.

Operations - Execute Tcl File in the menu bar, and select the target tcl file.

Method 4: directly drag the tcl file into the console window to execute the instructions in the -tcl

file, as shown in Figure 4 74.



Figure 4 -74 Drag the tcl file into the console window for execution

2. Execute under the Windows cmd command line

In the Windows environment, you can also use cmd to call cdt_cfg.exe to execute the tcl file, as shown in Figure 475 -. The following parameter settings are supported:

[-file] directory to be parsed *.tcl

[-args] is used to sequentially replace the data of the %s placeholder in the tcl file. Different data are separated by spaces and surrounded by curly braces.

Usage: When calling cdt_cfg.exe under cmd, users are supported to use %s as a placeholder in the (*.tcl) file to replace some parameters or instructions in the tcl file. Then the user enters -file *.tcl -args { data1 data2... datan } in cmd to execute the (*.tcl) file. In this way, the data in the curly braces will replace the %s placeholders in the tcl file in sequence, and the data in the curly braces must be separated by spaces.

Example usage:

Suppose there is a command statement cfg_add_device -file %s in the test.tcl file

Call cdt_cfg.exe under cmd, and enter -file test.tcl -args { D:/a.sbit }

You can replace the %s placeholder in the test.tcl file with D:/a.sbit.



Figure 4 -75 Use cmd to call cdt_cfg.exe under Windows

3. Execute in the terminal mode of Configuation

Start " cdt_cfg_shell.exe ", and execute Tcl commands or tcl files in the terminal mode of Configuation. For details, see Chapter 4.6.

4.4.6 Example of using Tcl

The figure below is an example of the use of Tcl commands in the tcl file, and explains each part of the Tcl commands.



Step 1 : Connect to the JTAG server, which is a necessary step before operating the JTAG server. Among them, -ip and -port are connected to the local computer by default. If connecting to the JTAG server of the local computer, keep the above default values.

Step 2: Add the target device and specify the target file. The target file needs to be replaced with the actual file of the customer (if multiple devices are added, the device_index of the last added device is 0 and is close to TDI).

Step 3 : The optional execution operation is determined according to the specific operation to be performed by the user. The above example is the operation of burning the temporary key file key.nky in the chip and burning the bit stream file test.sbit encrypted with this key. The key file needs to be replaced with the actual key file name of the user.

Step 4 : Disconnect the JTAG server connection and end the operation.

Note: Steps 1, 2, and 4 are necessary routine steps, and Step 3 is the operation that the user needs to perform each time.

4.5 Tcl command specific parameter query

Currently, the Configuration tool supports the following Tcl commands customized by our company.

4.5.1 Explanation and Usage of Common Commands

1.1 source

The source command is a built-in command of Tcl, which is used to execute Tcl scripts.

Usage: source C:/Users/Test/Desktop/test.tcl

1.2 cfg_help

cfg_help, lists all currently supported custom Tcl commands. no parameters

Usage: cfg_help

1.3 cfg_connect

cfg_connect, the command used to connect to the JtagServer server, contains 2 parameters -ip, -port

Usage: cfg_connect -ip 192.29.103.191 -port 65420

1.4 cfg_disconnect

cfg_disconnect, the command used to disconnect from the JtagServer server, no parameters

Usage: cfg_disconnect

1.5 cfg_set_delay_time

cfg_set_delay_time, Fabric Configuration delay command, in seconds (seconds), this command will cause CFG to delay and wait for the set time. Contains 1 parameter -time.

Usage: cfg_set_delay_time -time 3

1.6 cfg_close

cfg_close, close the Fabric Configuration software, no parameters

Usage: cfg_close

1.7 cfg_set_cable_property

cfg_set_cable_property, set the cable parameters connected to the current Jtag Server, including 2 parameters.

-index : Select the serial number of the cable to be operated

-freq: Set the frequency value of the cable, the current supported frequency values are : {15 MHz, 10 MHz, 7.5 MHz, 6 MHz,5 MHz,2 MHz, 1 MHz,500 KHz,300 KHz, 100 KHz}.

Usage: cfg_set_cable_property -index 0 -freq 10Mhz: means to set the frequency of the cable whose serial number is 0 to 10MHz.

1.8 cfg_set_tcl_break

cfg_set_tcl_break, set whether Tcl encounters an error and execute it down (execute Tcl file)

Usage: cfg_set_tcl_break -flag true indicates that Tcl will not continue to execute when encountering an error interrupt.

1.9 cfg_set_options

cfg_set_options, set the running properties of Jtag Server, a parameter

\- report_detail Set whether to write detailed reports on the board automation, parameter type Boolean_value (TRUE/FALSE)

[-server_delay_time] Set the delay time of the server, that is, how long the server starts to read data after the delay. The parameter type is an integer greater than 0

Usage: cfg_set_options -rpt_detail true -server_delay_time *integer-value*

4.5.2 Commands for FPGA Devices

Including the read function of FPGA ID, user code and status register, Tcl commands such as logic bit stream download, read back and check.

2.1 cfg_scan_chain

cfg_scan_chain, Scan Jtag chain command (initialize chain operation) no parameters

Usage: cfg_scan_chain

2.2 cfg_assign_file

cfg_assign_file, Configure the bitstream file command, which contains 2 parameters

-f ile Path to configuration bitstream file

[-device_index] specifies the device index, optional, the default is 0

Usage: cfg_assign_file -file D:/bs/counter.sbit -device_index 0

2.3 cfg_set_device_property

cfg_set_device_property, set device programming, readback, embedded Flash properties, including 4 parameters

[-device_index] specifies the device index, optional, the default is 0

[-program] specifies the programming attribute, which is a hexadecimal number

Setting example: If Bit0 and Bit1 are set to 1 at the same time, the program value is 3

Note: For the Compact series, Bit1 and Bit2 cannot be set to 1 at the same time

Bit0 : shutdown configuration (Compact \Logos\Logos2\Titan2)

Bit1 : ISC configuration (Compact\Logos\Logos2\Titan2)

Bit2 : Offline configuration (Compact)

[-readback] specifies the readback attribute, which is a hexadecimal number

Setting example: If Bit0 and Bit1 are set to 1 at the same time, the readback value is 3

Note: For the Compact series, Bit1 and Bit2 cannot be set to 1 at the same time

Bit0 : Turn off readback (Compact \Logos\Logos2\Titan2)

Bit1 : ISC readback (Compact\Logos\Logos2\Titan2)

Bit2 : offline readback (Compact)

[-eflash] Specifies the embedded Flash attribute, which is a hexadecimal number

Setting example: If only Bit0 and Bit6 are set to 1 at the same time, the eflash value is 41

Settings:

① Only set the Bit0-Bit7 attribute, then the software will automatically complete the Bit8-Bit51 attribute value

② There are two ways to set Bit0-Bit51 attributes at the same time. The first is to set the specified eflash attribute through the interface. After clicking OK, the Tcl console will print out the attribute value. For the specific attribute setting operation, see 4.3.19 ; the second is to use the function of recording user operations, first set the device attribute, programming the device, you can generate a tcl script, open the script file to get the tcl command setting properties, see 4.3.22 for details

Note: The default value of eflash is 4fd004fc000e6, only the Compact series supports this property setting

Bit0 : Indicates whether to enable erasing EFlash before downloading the bit stream

Bit1 : Indicates whether to enable verification of EFlash after downloading the bit stream

Bit2 : Indicates whether to erase the entire chip when erasing Eflash

Bit3 : Indicates whether to enable reset after erasing EFlash

Bit4 : Indicates whether reset is enabled after programming EFlash

Bit5 : Indicates whether reset is enabled after the programming feature control bit

Bit6 : Indicates whether to enable the programming feature control bit after downloading the bit stream

Bit7 : Indicates whether to enable the programming feature control bit before programming CRAM

Bit19-Bit8 : Operate the start address page of Embed Flash, used to set the start address page of programming, erasing and readback

Bit31-Bit20 : Operate the end address page of Embed Flash, used to lock the content of Embed flash from the beginning to the specified page

Bit51-Bit32 : Specify the size of the content read back from the start address page of the Embed

Flash operation, the unit is Byte

Usage: cfg_set_device_property -device_index integer-value -program hex-value -readback hex-value -eflash hex-value

2.4 cfg_program

cfg_program, programming bit stream command, contains 2 parameters

[-device_index] specifies the device index, optional, the default is 0

Usage: cfg_program -device_index 0

2.5 cfg_program_key_file

cfg_program_key_file, programming bit stream encryption key command, including 3 parameters

[-device_index] specifies the device index, optional, the default is 0

[-key_location] specifies the key storage location, the default is the temporary key register;

0: Temporary Key Register; 1: eFuse Register; 2: Battery Backed RAM

–file: specify the path information of the key file

Usage: cfg_program_key_file -device_index 0 -key_location 0 -file D:/bs/led.nky

2.6 cfg_readback

cfg_readback, command to read back bitstream file, contains 2 parameters

-file: specify the path information of the readback file

[-device_index] specifies the device index, optional, the default is 0

Usage: cfg_readback -file D:/bs/test.sbit -device_index 0

2.7 cfg_verify

cfg_verify, verify bitstream file command, contains 1 parameter

[-device_index] specifies the device index, optional, the default is 0

Usage: cfg_verify -device_index 0

2.8 cfg_read_device_property

cfg_read_device_property, get device property information, including 2 parameters

-mode e Get the mode of device attributes, that is, which attribute to get

0: get device ID

1: Get device UID

2: Get device UserCode

3: Get device status register

4: Read the value of the device EFuse

[-device_index] specifies the device index, optional, the default is 0
usage:
Read ID: cfg_read_device_property -mode 0 -device_index 0
Read UID: cfg_read_device_property -mode 1 -device_index 0
Read UserCode: cfg_read_device_property -mode 2 -device_index 0 Read Status Register : cfg_read_device_property -mode 3 -device_index 0

2.9 cfg_reset_fpga

cfg_reset_fpga, soft reset the FPGA device, 1 parameter
[-device_index] specifies the device index, optional, the default is 0
Usage: cfg_reset_fpga -device_index *integer-value*

2.10 cfg_operate_ctl

cfg_operate_ctl, operate the feature control bits of Compact series devices, 3 parameters;

-mode Operation characteristic control bit mode (0: program, 1: read only, 2: erase).

[-value] When the operation mode is programming, it is a mandatory parameter, and the input hexadecimal value is the downloaded characteristic value. The input value range is 0x00000000 – 0xFFFFFFFF.

[-device_index] specifies the device index, optional, the default is 0
Usage: cfg_operate_ctl -mode *integer-value* -value *hex-value*
-device_index *integer-value*

2.11 cfg_operate_eflash

cfg_operate_eflash, operate the embedded Flash of Compact series, 3 parameters;

-mode is the operation mode of the embedded Flash (0: sleep, 1: wake up, 2: lock, 3: program, 4: read-only, 5: erase, 6: verify).

[-file] Operate the sbit file required by the embedded Flash, download the bit stream file or read back the file

[-device_index] specifies the device index, optional, the default is 0

Usage: cfg_operate_eflash -mode *integer-value* -device_index *integer-value* -file E:/pan/eflash.sbit

2.12 cfg_read_register

cfg_read_register, read the contents of the specified address register. Contains 2 parameters,

-address (the address of the specified register)
[-device_index] specifies the device index, optional, the default is 0
usage: cfg_read_register -address *123* -device_index *0*

2.13 cfg_write_register

cfg_write_register, write the contents of the specified address register. Contains 3 parameters,

-address (the address of the specified register)

-value (given the value to write to the specified register)

[-device_index] specifies the device index, optional, the default is 0

Usage: cfg_write_register –address *123* –value *123* -device_index *0*

2.14 cfg_add_device

cfg_add_device, add device command. Contains 1 parameter,

-file configuration file path

Usage: cfg_add_device -file D:/bs/counter.sbit

2.15 cfg_remove_device

cfg_remove_device, delete the specified device command. Contains 1 parameter,

[-device_index] specifies the device index, optional, the default is 0

Usage: cfg_remove_device -device_index 0

2.16 cfg_remove_devices

cfg_remove_devices, the command to delete multiple specified devices. Contains 1 parameter, -device_index (multiple device index sets to be deleted)

Usage: cfg_remove_devices -device_index {0 1}

2.17 cfg_remove_all_devices

cfg_remove_all_devices, delete all devices command, no parameters.

Usage: cfg_remove_all_devices

2.18 cfg_program_user_efuse

cfg_program_user_efuse, Program 32 -bit user eFuse register command. Contains 2 parameters,

-value (given the value to write to the eFuse register, 32 -bit binary number)

[-device_index] specifies the device index, optional, the default is 0

Usage: cfg_program_user_efuse –value *000111* -device_index *0*

4.5.3 Commands for Flash devices

Tcl commands under this set mainly perform operations such as downloading, deleting, reading back, and verifying the Flash device on the board through the SPI interface.

3.1 cfg_flash_scan_device

cfg_flash _scan _device, scan SPI flash device command.

Usage: cfg_flash_scan_device

3.2 cfg_flash_assign_file

cfg_flash_assign_file, configure the flash bit stream file command, including 1 parameter,

-file (file to configure)

Usage: cfg_flash_assign_file -file D:/source/bit/counter_16.sfc

3.3 cfg_flash_program

cfg_flash_program, download flash bit stream file command, no parameters.

usage: cfg_flash_program

3.4 cfg_flash_erase

cfg_flash_erase, command to erase flash bitstream file, 1 parameter.

[-mode integer_number] Set the erase mode of Flash

0: auto selected Mode by file size 1: Block Erase 2: Chip Erase

3: Sector Erase 4: Page Erase

usage: cfg_flash_erase –mode integer_number

3.5 cfg_flash_readback

cfg_flash_readback, command to read back flash bit stream file, contains 1 parameter,

-file (the file to read back the saved bitstream from)

Usage: cfg_flash_readback -file D:/source/bit/testLUT5_readback.sfc

3.6 cfg_flash_verify

cfg_flash_verify, verify the flash bit stream file, no parameters.

Usage: cfg_flash_verify

3.7 cfg_flash_remove_device

cfg_flash_remove_device, delete flash device command, no parameters.

Usage: cfg_flash_remove_device

3.8 cfg_flash_read_register

cfg_flash_read_register, read Flash register value, no parameters.

Usage: cfg_flash_read_register

3.9 cfg_flash_write_register

cfg_flash_write_register, write Flash register value, 1 parameter

-value string Given the value written to the specified register.

Usage: cfg_flash_write_register –value 0x1c

4.5.4 Commands for Jtag Flash devices

Tcl commands under this set mainly perform operations such as downloading, deleting, reading back, and verifying the Flash device on the board through the Jtag interface. Since the operation of Flash through the Jtag interface depends on the port of the FPGA, when executing these commands, it is necessary to ensure that the CFG has obtained the FPGA information on the board, that is, the scan FPGA chain operation has been performed.

4.1 cfg_jtag_flash_scan_device

cfg_jtag_flash_scan_device, scan external Flash device, 1 parameter

[-device_index] specifies the device index, optional, the default is 0

usage: cfg_jtag_flash_scan_device –device_index 0

4.2 cfg_jtag_flash_assign_file

cfg_jtag_flash_assign_file, configure bit stream file sfc for external Flash, 2 parameters

--file configures the path to the bitstream file

[-device_index] specifies the device index, optional, the default is 0

Usage: cfg_jtag_flash_assign_file -file D:/bs/counter.sfc -device_index 0

4.3 cfg_jtag_flash_erase

cfg_jtag_flash_erase, erase external Flash device, 1 parameter

[-device_index] specifies the device index, optional, the default is 0

Usage: cfg_jtag_flash_erase –device_index 0

4.4 cfg_jtag_flash_program

cfg_jtag_flash_program, programming external Flash device, 1 parameter

[-device_index] specifies the device index, optional, the default is 0

Usage: cfg_jtag_flash_program –device_index 0

4.5 cfg_jtag_flash_verify

cfg_jtag_flash_verify, verify the bit stream configured by the external Flash device, 1 parameter

[-device_index] specifies the device index, optional, the default is 0

Usage: cfg_jtag_flash_verify -device_index 0

4.6 cfg_jtag_flash_readback

cfg_jtag_flash_readback, read back the bit stream configured by the external Flash device, 2 parameters

[-device_index] specifies the device index, optional, the default is 0

--file path to read back bitstream file

Usage: cfg_jtag_flash_readback –device_index 0

4.5.5 For on-board automation

A test case in the automated test on the board is a folder, and the file format contained must conform to any one of the following three format sets: { *.sbit , *.lcf , *.vcd} 、 {*.sbit , *.atp}, {*.sbit, *.avc}. The specific commands are as follows:

5.1 cfg_load_dir

cfg_load_dir, parse the test case directory to generate Tcl script, including 5 parameters

-file (directory to parse)

[-clock_mode] Set the clock for automation on the board, the default is the JTAG clock.

[-time] Set the delay time between program and load excite, the parameter type is an integer greater than 0

[-pkg] specifies the packaging information, the default is based on the bit file

[-scale_time] Specify the time range for generating *.vcd files

[-count] Set the number of times the test case is configured to cycle through the chip. The parameter range is 0 < count < 1000. If the parameter is greater than 1000, it is automatically set to 1000.

[-break] Set whether Tcl runs down when encountering an error, parameter type Boolean_value (TRUE/FALSE), true means that Tcl does not continue to execute when encountering an error

Usage: cfg_load_dir -file D:/auto_test –time integer-value indicates that the D:/auto_test folder and its subdirectories will be parsed, and the integer-value is delayed between program and load excite, in the file of each test case The corresponding TclScript.tcl script will be generated under the folder.

5.2 cfg_write_all_tcl_file

cfg_write_all_tcl_file, Collect the TclScript.tcl files in the specified

directory and its subdirectories to form a total AllTclScript.tcl containing a parameter

-file (directory to parse)

Usage: cfg_write_all_tcl_file -file D:/auto_test

5.3 cfg_load_excite

cfg_load_excite, load excite Contains 5 parameters,

-device_index (index of the device Default is 0)

–efile (vcd file)

–lfile (lcf file, or fdc file. The default is the file with the same name as vcd)

–rfile (report file name Default is report.txt)

-pfile (create a new folder for storage)

–clock_mode (clock mode, default is JTAG clock)

[-append] Set whether to save the report generated by multiple cycle configuration, parameter type Boolean_value (TRUE/FALSE), true means save

Usage: cfg_load_excite -efile D:/auto_test/test.vcd

5.4 cfg_write_all_report_file

cfg_write_all_report_file, collect all report files in the specified directory to form a general Excel file named AllReport.xlsx contains a parameter –file (directory to parse)

Usage: cfg_write_all_report_file –file D:/auto_test

5.5 cfg_write_auto_test

cfg_write_auto_test, generate an automated test script for the specified directory, Contains 5 parameters –file (directory to parse)

[-report_detail] Set whether to write a detailed report on the automation on the board, parameter type Boolean_value (TRUE/FALSE)

[-clock_mode] Set the clock for automation on the board, the default is the JTAG clock.

[-time] Set the delay time between program and load excite, the parameter type is an integer greater than 0

[-server_delay_time] Set the delay time of the server, that is, how long the server starts to read data after the delay. The parameter type is an integer greater than 0

[-count] Set the number of times the test case is configured to cycle through the chip. The parameter range is 0 < count < 1000. If the parameter is greater than 1000, it is automatically set to 1000.

[-break] Set whether Tcl runs down when encountering an error, parameter

type Boolean_value (TRUE/FALSE), true means that Tcl does not continue to execute when encountering an error

Usage: cfg_write_auto_test -file D:/auto_test -report_detail true -clock_mode 1 -server_delay_time *integer-value* –time *integer-value*

4.5.6 Convert file format

6.1 cfg_gen_sfc

cfg_gen_sfc, generate SFC files about flash, 5 parameters;

-sbit specifies the sbit file to convert

[-sfc] specifies the name of the generated sfc file, the default value is the sfc file with the same name under the same path as the sbit file

[-opcode] specifies the operation code, which is the read and write mode of Flash.

11(0x0B) : "Fast Read, 24-bit address"

59(0x3B) : "Dual Output Fast Read, 24-bit address"

107(0x6B) : "Quad Output Fast Read, 24-bit address"

12(0x0C) : "Fast Read, 32-bit address"

60(0x3C) : "Dual Output Fast Read, 32-bit address"

108(0x6C) : "Quad Output Fast Read, 32-bit address"

The default default value is 11 (0x0B).

[-device_name] specifies a compatible flash device, the default is N25Q256

[-is_x8] Convert to X8 data stream, the default is false.

[-sbit_start_address] Set the start position of the bit stream file

[-fast_mode] Set the chip to fast mode or non-fast mode (pgc series does not).

[-user_address_list] Set the starting address list of all user data

[-file_list] input file list

usage: cfg_gen_sfc –sbit sbit file

6.2 cfg_gen_chain_file

cfg_gen_chain_file, generate a cascade data stream download file. Contains two parameters,

-infile (specify the selected sbit file, separated by "," between files)

[-outfile] (specify the file to output the cascaded data stream, optional, the default is the first bit stream file name followed by _chain.sbit)

Usage: cfg_gen_chain_file -infile E:/pan/counter.sbit, E:/pan/led.sbit –outfile E:/pan/leSS.sbit

6.3 cfg_gen_multi_file

cfg_gen_multi_file, generate multiple data stream format files. For the meaning of specific parameters, please refer to Chapter 4 Fabric Configuration Software Description -> Section 20 device FPGA bit stream file conversion multiple format data streams.

-infile specifies the list of input files, that is, the files to be converted

-type specifies the format of the generated data stream

0: Multi-start data stream

1: SPI remote upgrade data flow

2: BPI remote upgrade data flow

3: Multifunctional data flow

4: Remote upgrade data flow

5: Master self-loading dual boot data flow

[-outfile] specifies the name of the output file, which is the same as the name of the first input file by default

[-sbit_start_address] Set the start address of the bit stream

[-insert_irst] Whether to insert hot start command when generating multi-start data stream

[-steup_index] specifies the boot application bitstream

[-svf_property] Set the svf property (hexadecimal / only for the main and self-loading dual-boot data stream), and the corresponding meanings of the hexadecimal bits are as follows.

1 : Enable Background ReConfig. default: Enable

2 : Enable Write Header and Comments default: Enable

3 : Enable SEC option of RUNTEST Command default:Enable

4 : Enable Split SVF into Function and Startup default:Disable

5 : Enable ISC Upgrade In Master Auto Config Mode default:Disable

6 : Enable Reset FPGA default:Enable

7: Enable Program Feature Control Value default:Disable

8: Enable Page Erase Embed Flash default:Disable

9: Enable Verify after Program Embed Flash

10 : Reset FPGA Command default:Enable

11 : Check Done of Status Register Command default:Disable

13 : Entire Erase Embed Flash Command default:Disable

14 : Enable Verify after Erase Embed Flash default:Disable

15 : Enable Verify after Program Embed Flash default:Disable

17 - 16 : Gen SVF Mode.(0:CRAM 1 : Embed Flash 2 : Command) default:1 :

Embed Flash

19 - 18 : SVF Frequency.(100KHz(0), 1MHz(1), 5MHz(2), 10MHz(3)) default: 1MHz

31 - 20 : SVF Start Page Address(Unit : 256Byte)

[-jtag_chain] manually set the jtag chain (hexadecimal / only for the main self-loading dual boot data stream

), the specific meanings are as follows. bit 0:7 previous device number

bit 8:15 in the later device number

bit 16:23 previous device command len

bit 24:31 in the later device command len

[-set_ctl] Whether to set the feature control bit (only for the main self-loading dual-boot data stream)

0 : no setting

1 : Use the feature control bit of the master self-loading mode

2 : Set the specific value of the feature control bit

[-split] Whether to generate the main self-loading data stream separately (only for the main self-loading dual boot data stream)

[-spi_mode] set spi mode

[-flatten] Whether to generate svf of SDR/SIR filling chain information.

6.4 cfg_svf_to_pef

cfg _svf_to_pef, compress the SVF according to a certain format to generate a PEF file.

-infile specifies the name of the input SVF file, that is, the file to be converted

[-outfile] specifies the name of the output file, which is the same as the name of the input file by default

[-compress] specifies whether to compress, the default compression

[-frequency] Specifies the frequency of rewriting the SVF file. If no parameter is provided, the default is the same as the original SVF file or 1 MHz

[-vendor] specifies the SVF file vendor, the default is the JTAG standard

6.5 cfg_one_step_create_svf

cfg_one_step_create_svf, generate svf files that can configure FPGA and embedded Flash. For the meaning of specific parameters, please refer to

Chapter 4 Fabric Configuration Software Instructions -> Section 19 Device Settings Operation Device Properties SVF Properties.

[-device_index] specifies the index of the operating device, optional, the default is the first device

[-svf_file_name] specifies the name of the generated SVF file, optional, the default is the same as the name of the bitstream file

[-svf_property] Set the svf property (hexadecimal), the corresponding meaning of the specific bits in hexadecimal is as follows.

1 : Enable Background ReConfig. default: Enable
2 : Enable Write Header and Comments default: Enable
3 : Enable SEC option of RUNTEST Command default: Enable
4 : Enable Split SVF into Function and Startup default: Disable
5 : Enable ISC Upgrade In Master Auto Config Mode default: Disable
6 : Enable Reset FPGA default: Enable
7: Enable Program Feature Control Value default: Disable
8: Enable Page Erase Embed Flash default:Disable
9: Enable Verify after Program Embed Flash
10 : Reset FPGA Command default:Enable
11 : Check Done of Status Register Command default:Disable
13 : Entire Erase Embed Flash Command default:Disable
14 : Enable Verify after Erase Embed Flash default:Disable
15 : Enable Verify after Program Embed Flash default:Disable
17 - 16 : Gen SVF Mode.(0:CRAM 1 : Embed Flash 2 : Command) default:1 : Embed Flash
19 - 18 : SVF Frequency.(100KHz(0), 1MHz(1), 5MHz(2), 10MHz(3)) default:1MHz
31 - 20 : SVF Start Page Address(Unit : 256Byte)

[-jtag_chain] Manually set the jtag chain (hexadecimal), the specific meaning is as follows.

bit 0:7 previous device number
bit 8:15 in the later device number
bit 16:23 previous device command len
bit 24:31 in the later device command len

[-ctl_value] This parameter is used to set the feature control bit. Note: If you want to set the value of the feature control bit, you must first set the 8th bit of svf_property to 1.

[-not_create_pef] set whether to generate pef file, the default is false

[-sbit] Manually select input bitstream file
[-flatten] Whether to generate svf of SDR/SIR filling chain information.

6.6 cfg_execute_svf

cfg_ execute **_svf,** execute SVF file or PEF file
-svf_file_name specifies the name of the input SVF or PEF file.
[-device_index] device location

6.7 cfg_create_chain_svf

cfg_create_chain_svf merges svf files
-infile input svf file path, separated by spaces, input at least two
[-outfile string] output file path

6.8 cfg_pef_to_svf

cfg_pef_to_svf converts pef files to svf files
-infile input pef file path
[-outfile] output file path

6.9 cfg_switch_flatten_svf

mutual conversion between ordinary svf and chained svf
-file input svf file path
[-outfile] output file path
[-mode] conversion mode, 0 : ordinary svf converted to chained svf
1 : Convert chained svf to normal svf

4.6 Terminal Mode Introduction

To use the Fabric Configuration function in a non-graphical interface system, you can use the terminal version of Fabric Configuration. Start the software -by double-clicking "cdt_cfg_shell.exe" in the bin directory under the installation directory of the PDS software. After startup, the interface is shown in Figure 4.76.

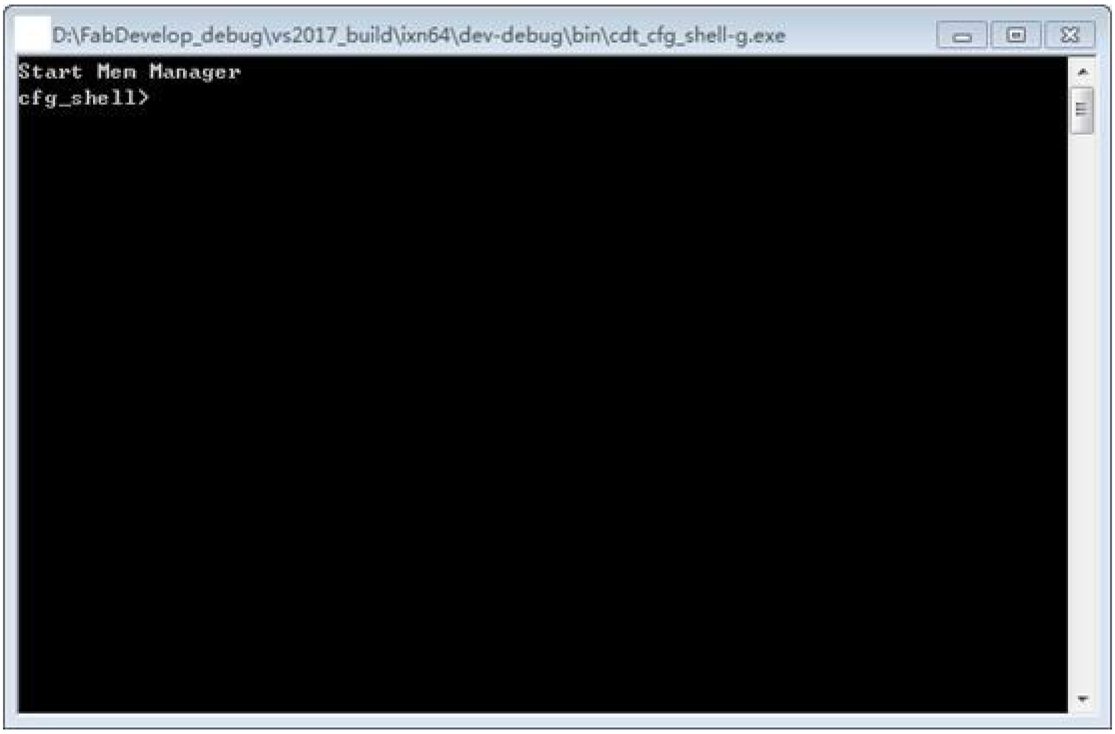


Figure 4 -76 cfg_shel display

The terminal mode uses the relevant functions through the tcl command, and the specific tcl functions can be found in Section 4.4. You can enter exit and cfg_close commands when exiting the software..

5. Appendix

5.1 The function represented by each bit of the instruction register

5.1.1 Titan series instruction register description

Table 8 Titan series instruction register description

<table>
<tr><th>bit</th><th>Name</th><th>Description</th></tr>
<tr><td>[9]</td><td>reserve</td><td>value is 0</td></tr>
<tr><td>[8]</td><td>init_b</td><td></td></tr>
<tr><td>[7]</td><td>efuse_prog_over</td><td>Indicates the end of eFuse programming</td></tr>
<tr><td>[6]</td><td>wakedown_over</td><td>Indicates wake-up shutdown complete</td></tr>
<tr><td>[5]</td><td>wakeup_over</td><td>Indicates that the wakeup is complete</td></tr>
<tr><td>[4]</td><td>init</td><td></td></tr>
<tr><td>[3]</td><td>isc_enabled</td><td></td></tr>
<tr><td>[2]</td><td>isc_done</td><td></td></tr>
<tr><td>[1]</td><td colspan="2" rowspan="2">It is a fixed value of 01, which realizes the detection and fault location of the integrity of the scanning link.</td></tr>
<tr><td>[0]</td></tr>
</table>

5.1.2 Logos series instruction register description

Table 9 Description of Logos series instruction register

<table>
<tr><th>bit</th><th>Name</th><th>Description</th></tr>
<tr><td>[9]</td><td>reserve</td><td>value is 0</td></tr>
<tr><td>[8]</td><td>efuse_prog_over</td><td>Indicates the end of eFuse programming</td></tr>
<tr><td>[7]</td><td>wakedown_over</td><td>Indicates wake-up shutdown complete</td></tr>
<tr><td>[6]</td><td>wakeup_over</td><td>Indicates that the wakeup is complete</td></tr>
<tr><td>[5]</td><td>init_b</td><td></td></tr>
<tr><td>[4]</td><td>init</td><td></td></tr>
<tr><td>[3]</td><td>isc_enabled</td><td></td></tr>
<tr><td>[2]</td><td>isc_done</td><td></td></tr>
<tr><td>[1]</td><td colspan="2" rowspan="2">It is a fixed value of 01, which realizes the detection and fault location of the integrity of the scanning link.</td></tr>
<tr><td>[0]</td></tr>
</table>

5.1.3 Logos2 series instruction register description

Table 10 Logos2 series instruction register description

bit	Name	Description
[9]	reserve	value is 0
[8]	efuse_prog_over	Indicates the end of eFuse programming
[7]	wakedown_over	Indicates wake-up shutdown complete
[6]	wakeup_over	Indicates that the wakeup is complete
[5]	init_n	
[4]	init	init_complete
[3]	isc_enabled	
[2]	isc_done	
[1]		
[0]	It is a fixed value of 01, which realizes the detection and fault location of the integrity of the scanning link.	

5.1.4 Compact series instruction register description

Table 11 Description of Compact Series Instruction Registers

bit	Name	Description
[9]	reserve	value is 0
[8]	busy	Indicates that the embedded FLASH is busy
[7]	wakedown_over	Indicates wake-up shutdown complete
[6]	wakeup_over	Indicates that the wakeup is complete
[5]	init_n	
[4]	init	
[3]	isc_enabled	
[2]	isc_done	
[1]		
[0]	It is a fixed value of 01, which realizes the detection and fault location of the integrity of the scanning link.	

5.1.5 Titan2 series instruction register description

Table 12 Titan2 series instruction register description

bit	Name	Description
[9]	reserve	value is 0
[8]	efuse_prog_over	Indicates the end of eFuse programming
[7]	wakedown_over	Indicates wake-up shutdown complete
[6]	wakeup_over	Indicates that the wakeup is complete
[5]	init_n	
[4]	init	init_complete
[3]	isc_enabled	
[2]	isc_done	
[1]		
[0]	It is a fixed value of 01, which realizes the detection and fault location of the integrity of the scanning link.	

5.2 The function represented by each bit of the status register

5.2.1 Titan series status register description

Table 13 Titan series status register description

bit	Name	Description
[31:22]		reserve
[21]	flg_x8_pal	parallel x8 mode
[20]	flg_x16_pal	Parallel x16 mode
[19]	flg_x32_pal	parallel x32 mode
[18:16]	m[2:0]	mode selection
[15]	key_lock	KEY reading and writing prohibited
[14]	init_b	init_b
[13]	init_complete	Configuration complete and error indication
[12]	done_i	external done
[11]	wakeup_over	end of wake up
[10]	wakedown_over	Shutdown wake-up ends
[9]	glogen	glogen
[8]	done	done
[7]	gwen	gwen
[6]	grsn	grsn
[5]	go out	go out
[4]	overflow	Configure overflow indication
[3]	flg_abort_ipal	Internal parallel abort indication
[2]	flg_abort_pal	Parallel abort indication
[1]	crc_err	CRC detection result 0: CRC is correct 1: CRC is wrong
[0]	id_err	ID detection result 0: correct 1: wrong

5.2.2 Logos series status register description

Table 14 Logos series status register description

bit	Name	Description
[31:30]		reserve
[29]	over_temp	over temperature flag
[28]	flg_x32	32 -bit data width indication from parallel mode
[27]	flg_x16	16 -bit data width indication from parallel mode
[26]	flg_x8	8 -bit data width indication from parallel mode
[25:24]	ipal_m[1:0]	Internal parallel interface data width selection
[23]	key_lock	key lock sign
[22]	fallback	fallback indicator
[21]	dci_match	DCI matching flag
[20]	pll_lock	PLL lock flag
[19]	gwen	global write enable
[18]	grsn	Global Register Set Reset
[17]	go out	Global IO output enable
[16]	vddt_n	LDO switch enable
[15]	glogen_fb	Global Logic Enable Feedback
[14]	glogen	global logic enable

[13]	done_i	CFG_DONE pin input
[12]	done	Device Wakeup Successful Flag
[11]	init_b	init_flag_n pin input
[10]	init_complete	Initialization complete and configuration error indication
[9:7]	m[2:0]	mode selection
[6]	wakedown_over	Wake-up shutdown ends
[5]	wakeup_over	end of wake up
[4]	overflow	FIFO overflow
[3]	timeout	watchdog timeout
[2]	rbcrc_err	Read back CRC test result 0: CRC is correct 1: CRC is wrong
[1]	crc_err	CRC test result 0: CRC is correct 1: CRC is wrong
[0]	id_err	ID test result 0: correct 1: wrong

5.2.3 Logos2 series status register description

Table 15 Logos2 series status register description

bit	Name	Description
[31:30]		reserve
[29]	prcfg_over	Partial reconfiguration complete flag
[28]	prcfg_err	partial reconfiguration error flag
[27]	over_temp	over temperature flag
[26]	flg_x32	32 -bit data width indication from parallel mode
[25]	flg_x16	16 -bit data width indication from parallel mode
[24]	flg_x8	8 -bit data width indication from parallel mode
[23:22]	ipal_m[1:0]	Internal parallel interface data width selection
[21]	fallback	fallback indicator
[20]	dci_match	DCI matching flag
[19]	pll_lock	PLL lock flag
[18]	gwen	global write enable
[17]	grsn	Global Register Set Reset
[16]	go out	Global IO output enable
[15]	glogen_fb	Global Logic Enable Feedback
[14]	glogen	global logic enable
[13]	done_i	CFG_DONE pin input
[12]	done	Device Wakeup Successful Flag

[11]	init_n	INIT_N pin input
[10]	init_complete	Initialization complete and configuration error indication
[9:7]	m[2:0]	mode selection
[6]	wakedown_over	Wake-up shutdown ends
[5]	wakeup_over	end of wake up
[4]	timeout	watchdog timeout
[3]	rbcrc_err	Read back CRC test result 0: CRC is correct 1: CRC is wrong
[2]	aut_err	Certification result 0: Authentication passed 1: Authentication failed
[1]	crc_err	CRC test result 0: CRC is correct 1: CRC is wrong
[0]	id_err	ID test result 0: correct 1: wrong

5.2.4 Compact series status register description

Table 16 Compact series status register description

bit	Name	Description
[31:30]	reserve	
[29]	persist_mspi	SPI interface enable in user mode
[28]	persist_si2c	Enabled from the I 2 C interface in user mode
[27]	persist_sspi	SPI interface enable in user mode
[26]	persist_jtag	JTAG interface enable in user mode
[25]	persist_done	DONE pin enable in user mode
[24]	persist_init	INIT_N pin enable in user mode
[23]	persist_rstn	RST_N pin enable in user mode
[22]	pass_cal	Embedded FLASH calibration success flag This bit is reserved for 4K/7K devices
[21]	busy	Embedded FLASH busy flag
[20]	lock	Embedded FLASH lock flag

		After the embedded FLASH is locked, the main self-loading data stream and user FLASH cannot be programmed and read, only the main self-loading data stream and user FLASH can be erased.
[19]	wake up	Embedded FLASH wake-up flag
[18]	sleep	Embedded FLASH dormancy flag
[17]	fallback	fallback indicator
[16]	pll_lock	PLL lock flag
[15]	gwen	global write enable
[14]	grsn	Global Register Set Reset
[13]	go out	Global IO output enable
[12]	reserve	
[11]	glogen_fb	Global Logic Enable Feedback
[10]	glogen	global logic enable
[9]	done_i	DONE pin input
[8]	done	Device Wakeup Successful Flag
[7]	init_n	INIT_N pin input
[6]	init_complete	Initialization complete and configuration error indication
[5]	wakedown_over	Wake-up shutdown ends
[4]	wakeup_over	end of wake up
[3]	timeout	watchdog timeout
[2]	rbcrc_err	Read back CRC test result 0: CRC is correct 1: CRC is wrong
[1]	crc_err	CRC test result 0: CRC is correct 1: CRC is wrong
[0]	id_err	ID test result 0: correct 1: wrong

5.2.5 Titan2 series status register description

Table 17 Titan2 series status register description

bit	Name	Description
[31:30]		reserve

[29]	prcfg_over	Partial reconfiguration complete flag
[28]	prcfg_err	partial reconfiguration error flag
[27]	over_temp	over temperature flag
[26]	flg_x32	32 -bit data width indication from parallel mode
[25]	flg_x16	16 -bit data width indication from parallel mode
[24]	flg_x8	8 -bit data width indication from parallel mode
[23:22]	ipal_m[1:0]	Internal parallel interface data width selection
[21]	fallback	fallback indicator
[20]	dci_match	DCI matching flag
[19]	pll_lock	PLL lock flag
[18]	gwen	global write enable
[17]	grsn	Global Register Set Reset
[16]	go out	Global IO output enable
[15]	glogen_fb	Global Logic Enable Feedback
[14]	glogen	global logic enable
[13]	done_i	CFG_DONE pin input
[12]	done	Device Wakeup Successful Flag
[11]	init_n	INIT_N pin input
[10]	init_complete	Initialization complete and configuration error indication
[9:7]	m[2:0]	mode selection
[6]	wakedown_over	Wake-up shutdown ends
[5]	wakeup_over	end of wake up
[4]	timeout	watchdog timeout
[3]	rbcrc_err	Read back CRC test result 0: CRC is correct 1: CRC is wrong
[2]	aut_err	Certification result 0: Authentication passed 1: Authentication failed
[1]	crc_err	CRC test result 0: CRC is correct 1: CRC is wrong
[0]	id_err	ID test result

		0: correct 1: wrong

5.3 Description of the feature control bits of the Compact series

5.3.1 Feature Control Bits Supported by PGC 1K and 2K Series

Table 18 Feature control bits supported by PGC 1K and 2K series

bit	Name	Defaults	Description
[0]	Enable Master Auto Mode In Configuration	0: Disabled by default	Whether to enable the main self-loading mode 0: disable 1: enable
[1]	Disable RST_N Pin In Configuration Mode	1 : disabled by default	RST_N pin in configuration mode 0: not disabled 1: disabled
[2]	Enable INIT Pin In Configuration Mode	0: Disabled by default	Whether to enable the INIT pin in configuration mode 0: disable 1: enable
[3]	Enable Done Pin In Configuration Mode	0: Disabled by default	Whether to enable the DONE pin in configuration mode 0: disable 1: enable
[4]	Enable Master SPI Mode In Configuration	0 : Disabled by default	Whether to enable master SPI mode 0: disable 1: enable
[5]	Disable JTAG Mode In Configuration Mode	0 : not disabled by default	Whether to disable configuration mode from the Jtag interface 0: not disabled 1: disabled
[6]	Disable Slave SPI Mode In Configuration	0: not disabled by default	slave SPI interface in configuration mode 0: not disabled 1: disabled

[7]	Disable Slave I2C Mode In Configuration	0: not disabled by default	I2C interface in configuration mode 0: not disabled 1: disabled
[8]	Disable Slave Parallel Mode In Configuration	1: disabled by default	Whether to disable slave parallel interface in configuration mode 0: not disabled 1: disabled
[9]	Enable Double Start Up	0 : Disabled by default	Whether to enable dual boot mode 0: disable 1: enable
[10]	Enable Fast Master SPI Mode	0 : Disabled by default	Whether to enable master SPI mode 0: disable 1: enable
[11]	Disable Persist RST_N Pin In User Mode	1 : not disabled by default	RST_N pin in user mode 0: not disabled 1: disabled
[12]	Enable Persist INIT Pin In User Mode	0: Disabled by default	Whether to enable the INIT pin in user mode 0: disable 1: enable
[13]	Enable Persist Done Pin In User Mode	0: Disabled by default	Whether to enable the DONE pin in user mode 0: disable 1: enable
[14]	Disable Persist JATG Mode In User Mode	0 : not disabled by default	JTAG interface in user mode 0: not disabled 1: disabled
[15]	Disable Persist Slave SPI Mode In User Mode	0: not disabled by default	Whether to disable user mode slave SPI interface 0: not disabled 1: disabled

[16]	Disable Persist Slave I2C Mode In User Mode	0: not disabled by default	Whether to disable user mode slave I2C interface 0: not disabled 1: disabled
[17]	Disable Persist Slave Parallel Mode In User Mode	1: disabled by default	Whether to disable slave parallel interface in user mode 0: not disabled 1: disabled
[27:18]	I2C Address	The default is 2'b0000000000	I2C device address
[28]	Enable Persist Master SPI Mode In User Mode	0 : Disabled by default	Whether to enable the main SPI interface in user mode 0: disable 1: enable
[29]	Enable Slowly Read Mode	0 : Disabled by default	slow read enable 0: Master self-loading, read embedded FLASH instruction, read feature control bit instruction, and read trim instruction, the embedded FLASH interface adopts normal read mode 1: All embedded FLASH read operations use slow read mode
[31:30]	reserve		
	Note: The default value of the feature control bits of Compact 1K and 2K series is: 0x00020902		

5.3.2 Feature Control Bits Supported by PGC 4K, 7K, 10K Series

Table 19 Feature control bits supported by PGC 4K, 7K, 10K series

bit	name	Defaults	Description
[0]	Enable Master Auto Mode In Configuration	0: Disabled by default	Whether to enable the main self-loading mode

			0: disable 1: enable
[1]	Disable RST_N Pin In Configuration Mode	1 : disabled by default	RST_N pin in configuration mode 0: not disabled 1: disabled
[2]	Enable INIT Pin In Configuration Mode	0: Disabled by default	Whether to enable the INIT pin in configuration mode 0: disable 1: enable
[3]	Enable Done Pin In Configuration Mode	0: Disabled by default	Whether to enable the DONE pin in configuration mode 0: disable 1: enable
[4]	Enable Master SPI Mode In Configuration	0 : Disabled by default	Whether to enable master SPI mode 0: disable 1: enable
[5]	Disable JTAG Mode In Configuration Mode	0 : not disabled by default	Whether to disable configuration mode from the Jtag interface 0: not disabled 1: disabled
[6]	Disable Slave SPI Mode In Configuration	0: not disabled by default	slave SPI interface in configuration mode 0: not disabled 1: disabled
[7]	Disable Slave I2C Mode In Configuration	0: not disabled by default	I2C interface in configuration mode 0: not disabled 1: disabled
[8]	Disable Slave Parallel Mode In Configuration	1: disabled by default	Whether to disable slave parallel interface in configuration mode 0: not disabled 1: disabled
[9]	Enable Double Start Up	0 : Disabled by default	Whether to enable dual boot mode 0: disable 1: enable
[10]	Enable Fast Master SPI Mode	0 : Disabled by default	Whether to enable master SPI mode 0: disable 1: enable
[11]	Disable Persist RST_N Pin In User Mode	1 : not disabled by default	RST_N pin in user mode 0: disabled 1: not disabled
[12]	Enable Persist INIT Pin In User	0: Disabled by default	Whether to enable the INIT

	Mode		pin in user mode 0: disable 1: enable
[13]	Enable Persist Done Pin In User Mode	0: Disabled by default	Whether to enable the DONE pin in user mode 0: disable 1: enable
[14]	Disable Persist JATG Mode In User Mode	0 : not disabled by default	JTAG interface in user mode 0: not disabled 1: disabled
[15]	Disable Persist Slave SPI Mode In User Mode	0: not disabled by default	Whether to disable user mode slave SPI interface 0: not disabled 1: disabled
[16]	Disable Persist Slave I2C Mode In User Mode	0: not disabled by default	Whether to disable user mode slave I2C interface 0: not disabled 1: disabled
[17]	Disable Persist Slave Parallel Mode In User Mode	1: disabled by default	Whether to disable slave parallel interface in user mode 0: not disabled 1: disabled
[27:18]	I2C Address	The default is 2'b0000000000	I2C device address
[28]	Enable Persist Master SPI Mode In User Mode	0 : Disabled by default	Whether to enable the main SPI interface in user mode 0: disable 1: enable
[29]	Enable Slowly Read Mode	0 : Disabled by default	slow read enable 0: Master self-loading, read embedded FLASH instruction, read feature control bit instruction, and read trim instruction, the embedded FLASH interface adopts normal read mode 1: All embedded FLASH read operations use slow read mode
[30]	msdboot_en	0 : Disabled by default	Active self-loading dual boot enable
[31]	reserve		

Note: The default value of the feature control bits of Compact 4K, 7K, 10K series is: 0x00020902

5.4 Currently supported Flash devices and their matching read/write modes

Table 20 Currently supported Flash devices and their matching read/write modes

Flash device name	Supported read and write modes
M25P16	SPI X1, 24-bit address
M25P32	SPI X1, 24-bit address
M25P64	SPI X1, 24-bit address
M25P128	SPI X1, 24-bit address
N25Q32	SPI X1, 24-bit address; SPI X1, 24-bit address; SPI X1, 24-bit address
N25Q32	SPI X1, 24-bit address; SPI X1, 24-bit address; SPI X1, 24-bit address
N25Q64	SPI X1, 24-bit address; SPI X1, 24-bit address; SPI X1, 24-bit address
N25Q128	SPI X1, 24-bit address; SPI X1, 24-bit address; SPI X1, 24-bit address
N25Q256	SPI X1, 24-bit address; SPI X1, 24-bit address; SPI X1, 24-bit address SPI X1, 32-bit address; SPI X1, 32-bit address; SPI X1, 32-bit address
N25Q512	SPI X1, 24-bit address; SPI X1, 24-bit address; SPI X1, 24-bit address SPI X1, 32-bit address; SPI X1, 32-bit address; SPI X1, 32-bit address
GD25Q80C	SPI X1, 24-bit address, SPI X2, 24-bit address, SPI X4, 24-bit address
GD25Q32C	SPI X1, 24-bit address; SPI X1, 24-bit address; SPI X1, 24-bit address
GD25Q64C	SPI X1, 24-bit address; SPI X1, 24-bit address; SPI X1, 24-bit address
GD25Q128C	SPI X1, 24-bit address; SPI X1, 24-bit address; SPI X1, 24-bit address
GD25Q256D	SPI X1, 24-bit address; SPI X1, 24-bit address; SPI X1, 24-bit address SPI X1, 32-bit address; SPI X1, 32-bit address; SPI X1, 32-bit address
GD25Q512D	SPI X1, 32-bit address; SPI X2, 32-bit address; SPI X4, 32-bit address SPI X1, 24-bit address; SPI X2, 24-bit address; SPI X4, 24-bit address
MD25Q80C	SPI X1, 24-bit address, SPI X2, 24-bit address, SPI X4, 24-bit address
W25Q40CL	SPI X1, 24-bit address; SPI X1, 24-bit address; SPI X1, 24-bit address
W25Q80	SPI X1, 24-bit address; SPI X1, 24-bit address; SPI X1, 24-bit address
W25Q16	SPI X1, 24-bit address; SPI X1, 24-bit address; SPI X1, 24-bit address
W25Q32	SPI X1, 24-bit address; SPI X1, 24-bit address; SPI X1, 24-bit address
W25Q64	SPI X1, 24-bit address; SPI X1, 24-bit address; SPI X1, 24-bit address
W25Q128	SPI X1, 24-bit address; SPI X1, 24-bit address; SPI X1, 24-bit address
W25Q256	SPI X1, 24-bit address; SPI X1, 24-bit address; SPI X1, 24-bit address SPI X1, 32-bit address; SPI X1, 32-bit address; SPI X1, 32-bit address
W25Q512	SPI X1, 32-bit address, SPI X2, 32-bit address, SPI X4, 32-bit address SPI X1, 24-bit address, SPI X2, 24-bit address, SPI X4, 24-bit address
XM25QH16B	SPI X1, 24-bit address; SPI X1, 24-bit address; SPI X1, 24-bit address
XM25QH32B	SPI X1, 24-bit address; SPI X1, 24-bit address; SPI X1, 24-bit address
XM25QH64A	SPI X1, 24-bit address; SPI X1, 24-bit address; SPI X1, 24-bit address
XM25QH128A	SPI X1, 24-bit address; SPI X1, 24-bit address; SPI X1, 24-bit address
XM25QH256B	SPI X1, 32-bit address, SPI X2, 32-bit address, SPI X1, 24-bit address SPI X2, 24-bit address
SM25QH256M	SPI X1, 32-bit address; SPI X2, 32-bit address; SPI X1, 24-bit address SPI X2, 24-bit address
S25FL64	SPI X1, 24-bit address; SPI X2, 24-bit address; SPI X4, 24-bit address
S25FL128	SPI X1, 24-bit address; SPI X2, 24-bit address; SPI X4, 24-bit address

S25FL256L	SPI X1, 32-bit address; SPI X2, 32-bit address; SPI X4, 32-bit address SPI X1, 24-bit address; SPI X2, 24-bit address; SPI X4, 24-bit address
S25FL256S	SPI X1, 32-bit address; SPI X2, 32-bit address; SPI X4, 32-bit address SPI X1, 24-bit address; SPI X2, 24-bit address; SPI X4, 24-bit address
S25FL512	SPI X1, 32-bit address; SPI X2, 32-bit address; SPI X4, 32-bit address SPI X1, 24-bit address; SPI X2, 24-bit address; SPI X4, 24-bit address
ZB25VQ16	SPI X1, 24-bit address; SPI X1, 24-bit address; SPI X1, 24-bit address
ZB25VQ32	SPI X1, 24-bit address; SPI X2, 24-bit address
MX25L32	SPI X1, 24-bit address; SPI X2, 24-bit address
MX25L64	SPI X1, 24-bit address; SPI X2, 24-bit address
MX25L128	SPI X1, 24-bit address; SPI X2, 24-bit address
MX25L256	SPI X1, 32-bit address; SPI X2, 32-bit address
MX25L512	SPI X1, 32-bit address; SPI X2, 32-bit address; SPI X1, 24-bit address SPI X2, 24-bit address
IS25LP16D	SPI X1, 24-bit address; SPI X2, 24-bit address
IS25LP32D	SPI X1, 24-bit address; SPI X2, 24-bit address
IS25LP64D	SPI X1, 24-bit address; SPI X2, 24-bit address
IS25LP128D	SPI X1, 32-bit address; SPI X2,32-bit address; SPI X1,24-bit address SPI X2, 24-bit address
IS25LP256D	SPI X1, 32-bit address; SPI X2, 32-bit address; SPI X1, 24-bit address SPI X2, 24-bit address
IS25LP512D	SPI X1, 32-bit address; SPI X2, 32-bit address; SPI X1, 24-bit address SPI X2, 24-bit address
IS25WP16D	SPI X1, 24-bit address; SPI X2, 24-bit address
IS25WP32D	SPI X1, 24-bit address; SPI X2, 24-bit address
IS25WP64D	SPI X1, 24-bit address; SPI X2, 24-bit address
IS25WP128D	SPI X1, 32-bit address; SPI X2, 32-bit address; SPI X1, 24-bit address SPI X2, 24-bit address
IS25WP256D	SPI X1, 32-bit address; SPI X2, 32-bit address; SPI X1, 24-bit address SPI X2, 24-bit address
IS25WP512D	SPI X1, 32-bit address; SPI X2, 32-bit address; SPI X1, 24-bit address SPI X2, 24-bit address
EN25Q80C	SPI X1, 24-bit address; SPI X2, 24-bit address
AT45DB081E	SPI X1, 24-bit address; SPI X2, 24-bit address
FH25VQ80D	SPI X1, 24-bit address; SPI X2, 24-bit address

6. Contacts

Контакты для технических и коммерческих вопросов

ООО «Инэк»

г. Санкт-Петербург, ул. Яблочкова, д. 20, литер А, оф. 504

contact@innek.ru

+7 (812) 200-40-37